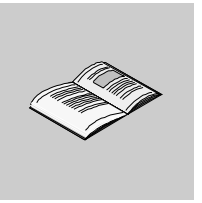
Concept 2.6
Bibliothèque de blocs IEC
Intercalaire : FUZZY

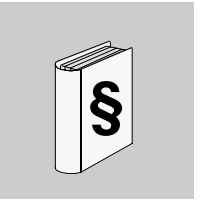
01/2007

33002228.03

Table des matières



Consignes de sécurité



Informations importantes

AVIS

Veuillez lire soigneusement ces consignes et examiner l'appareil afin de vous familiariser avec lui avant son installation, son fonctionnement ou son entretien. Les messages particuliers qui suivent peuvent apparaître dans la documentation ou sur l'appareil. Ils vous avertissent de dangers potentiels ou attirent votre attention sur des informations susceptibles de clarifier ou de simplifier une procédure.

L'apposition de ce symbole à un panneau de sécurité Danger ou Avertissement signale un risque électrique pouvant entraîner des lésions corporelles en cas de non-respect des consignes.

Ceci est le symbole d'une alerte de sécurité. Il vous avertit d'un risque de blessures corporelles. Respectez scrupuleusement les consignes de sécurité associées à ce symbole pour éviter de vous blesser ou de mettre votre vie en danger.

⚠ DANGER

DANGER indique une situation immédiatement dangereuse qui, si elle n'est pas évitée, **entraînera** la mort ou des blessures graves.

⚠ AVERTISSEMENT

AVERTISSEMENT indique une situation présentant des risques susceptibles de **provoquer** la mort, des blessures graves ou des dommages matériels.

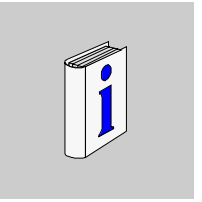
⚠ ATTENTION

ATTENTION indique une situation potentiellement dangereuse et susceptible d'**entraîner** des lésions corporelles ou des dommages matériels.

REMARQUE IMPORTANTE

Les équipements électriques doivent être installés, exploités et entretenus par un personnel d'entretien qualifié. Schneider Electric n'assume aucune responsabilité des conséquences éventuelles découlant de l'utilisation de cette documentation.

A propos de ce manuel



Présentation

Objectif du document

Cette documentation vous aidera à configurer les fonctions et les blocs fonction.

Champ d'application

Cette documentation s'applique à la version 2.6 de Concept pour Microsoft Windows 98, Microsoft Windows Version 2000, Microsoft Windows XP ou Microsoft Windows NT 4.x.

Note : Vous trouverez d'autres Notas à jour dans le fichier README.WRI de Concept.

Historique des évolutions

Indice	Liste des évolutions
1	Concept 2.6 - FUZZY Library
2	Concept 2.6 SR2 - FUZZY Library (Version 2 cause of deleted logo in Bib-info)
3	Concept 2.6 SR4 Patch A - FUZZY Library (OPR 20037547)

Document à consulter

Titre	Référence
Instructions d'installation de Concept	840 USE 502 01
Manuel utilisateur de Concept	840 USE 503 01
Concept EFB User Manual	840 USE 505 00
Bibliothèque de blocs LL984 de Concept	840 USE 506 01

Vour pouvez télécharger ces publications techniques ainsi que d'autres informations techniques à partir de notre site Web : *www.telemecanique.com*

Avertissements liés au(x) produit(s)

Commentaires utilisateur

Envoyez vos commentaires à l'adresse e-mail techpub@schneider-electric.com

Généralités sur la bibliothèque du bloc FUZZY

I

Résumé

Introduction

Ce chapitre contient des informations générales sur la bibliothèque du bloc FUZZY.

Contenu de cette partie

Cette partie contient les chapitres suivants :

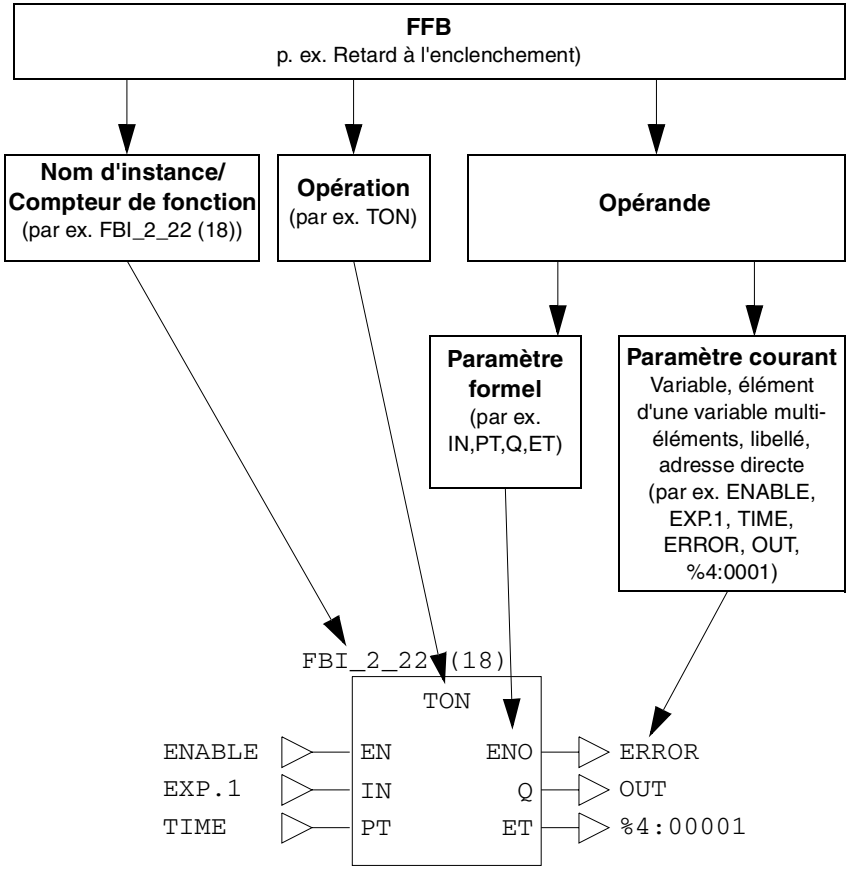
Paramétrage des fonctions et blocs fonction

1

Paramétrage des fonctions et blocs fonction

Généralités

Tout FFB se compose d'une opération, des opérandes nécessaires à l'opération et d'un nom d'instance/numéro de fonction.



Opération

L'opération détermine la fonctionnalité qui doit être exécutée par le FFB, p. ex. registre à décalage ou opérations de conversion.

Opérande

L'opérande détermine avec quoi l'opération doit être exécutée. Dans les FFB, il est constitué de paramètres formels et de paramètres réels.

Paramètre formel/paramètre réel

Le paramètre formel réserve la place pour un opérande. Lors du paramétrage, un paramètre actualisé (paramètre réel) est affecté au paramètre formel.

Le paramètre réel peut être une variable, une variable multi-éléments, un élément d'une variable multi-éléments, un libellé ou une adresse directe.

Lancement conditionnel/ inconditionnel

Chaque FFB peut disposer d'un lancement "conditionnel" ou "non conditionnel". La condition est réalisée par une connexion préalable de l'entrée EN.

- EN démasqué
  appel conditionnel (le FFB est traité uniquement lorsque EN = 1)
- EN masqué
  appel non conditionnel (le FFB est toujours traité)

Note : Si elle n'est pas paramétrée, l'entrée EN doit être masquée. Étant donné que les entrées non paramétrées sont automatiquement occupées par un "0", le FFB ne serait jamais exécuté.

Note : Dans le cas des blocs fonction bloqués (EN = 0) disposant d'une fonction temporelle interne (par exemple, DELAY), il semble que le temps continue de s'écouler, car il est calculé à l'aide de l'horloge système, le rendant indépendant du cycle programme et de la validation du bloc.

Appel de fonctions et DE blocs fonction en IL et ST

Pour l'appel des fonctions et des blocs fonction dans IL (liste d'instructions) et ST (littéral structuré), veuillez vous référer aux chapitres correspondants du manuel de l'utilisateur.

Fuzzy-Control

2

Résumé

Introduction

Ce chapitre explique d'abord plus en détail ce que le Fuzzy-Control représente, et comment le Fuzzy-Control peut être mis en uvre pour des opérations de règlage et de commande. Pour la mise en uvre du Fuzzy-Control avec ConCept, les fonctions élémentaires et les blocs fonctions (EFB) sont mis à disposition, leur fonctionnement conjoint est également expliqué plus en détail dans ce chapitre.

Contenu de ce chapitre

Ce chapitre contient les sous-chapitres suivants :

2.1 Introduction à la théorie du Fuzzy-Control

Résumé

Introduction

Cette section donne un aperçu des bases du Fuzzy-Control

Note : Ce chapitre ne tente pas de décrire le Fuzzy-Control de manière mathematiquement exacte, mais de transmettre la compréhension du Fuzzy-Control. Vous trouverez de plus amples informations sur la théorie dans la littérature correspondante.

Contenu de ce sous-chapitre

Ce sous-chapitre contient les sujets suivants :

Bases du Fuzzy-Control

Introduction

La notion "Fuzzy-Logic'' décrit un théorie des "quantités floues'".

Dans le domaine scientifique ou dans les programmes informatiques, on travaille habituellement avec seulement deux valeurs - vrai ou faux, marche ou arrêt, 1 ou 0. Il n'existe pas de valeurs intermédiaires.

La ligne maîtresse de la Fuzzy-Logic, par contre, est le rapport avec des quantités floues, donc avec des quantités, dont les éléments n'appartiennent que progressivement à une quantité. Au lieu de n'autoriser que "associé'" et "non associé", on autorise également des états intermédiaires. A l'aide de la Fuzzy-Logic, on peut ainsi mathématiquement décrire et manipuler des incertitudes de tout type.

Cette théorie trouve son utilisation la plus efficace en technique de régulation avec le Fuzzy-Control (régulateur Fuzzy).

Exemple : Température

Si l'on considère une valeur physique, comme la température, alors cette valeur est en général représentée par un nombre et par une unité physique, donc par exemple 21 degrés C.

La température peut toutefois être tout aussi bien décrite de manière verbale par attributs. Ainsi la température peut être décrite par les attributs "froid", "agréable" et "chaud", si nous parlons p.ex. de la température ambiante. Ces attributs ne sont plus exactement délimités l'un par rapport à l'autre, ce sont des descriptions floues de la température.

Une température ambiante de 15 degrés C est généralement qualifiée de "froide", tandis qu'une température de 21 degrés C l'est de "chaude". Si l'on se réfère à la sensation humaine, il n'existe pas une stricte ligne de partage entre "froid" et "agréable", comme on la connaît dans la logique classique, qui ne travaille, il est vrai, qu'avec les notions logiques "vrai" (TRUE, 1) ou "faux" (FALSE, 0). Il s'agit bien plus d'une transition de "froid" à "agréable".

Fuzzy-Control pour la technique de régulation/contrôle

Introduction

Pour la technique de régulation/contrôle, on produit une ou plusieurs sorties réponses dépendantes d'une ou plusieurs valeurs d'entrée. Les valeurs d'entrée sont reliées entre elles de manière logique et numérique. Le résultat de la liaison ou du calcul sont des valeurs de sortie. Le Fuzzy-Control procède de même.

La valeur d'entrée est transmise en une variable linguistiqueà l'aide de termes linguistiques et enfin évaluée à l'aide de règles. Le résultat, également une variable linguistique "floue", doit alors être à nouveau transmis en une sortie réponse, étant donné qu'avec une variable "floue" aucune valve ne se laisse diriger.

Le Fuzzy-Control dans la technique de règlage et de commande

Déroulement dans la technique de règlage et de commande

Echelon	Description
1	Les valeurs d'entrées sont fuzzifiées
2	Les termes linguistiques obtenus sont reliés entre eux selon des règles déterminées avec les opérateurs correspondants. Résultat du traitement de toutes les règles (inférence) : une variable de sortie "floue", c'est-à-dire une variable, qui est décrite par des attributs de ses degrés d'appartenance.
3	Defuzzification des variables de sortie en un nombre univoque, auquel, dans le processus de règlement, une action puisse être associée.

Notions de la théorie Fuzzy

Variables linguistiques et termes linguistiques

Exemple

Si la température est définie comme une variable linguistique , celle-ci peut alors être décrite par les termes linguistiques "froid", "agréable" et "chaud".

Définition

- Une variable linguistique est une variable (p. ex. température), qui est décrite par des termes linguistiques.
- Des termes linguistiques décrivent des attributs d'une variable linguistique.

Degré d'appartenance

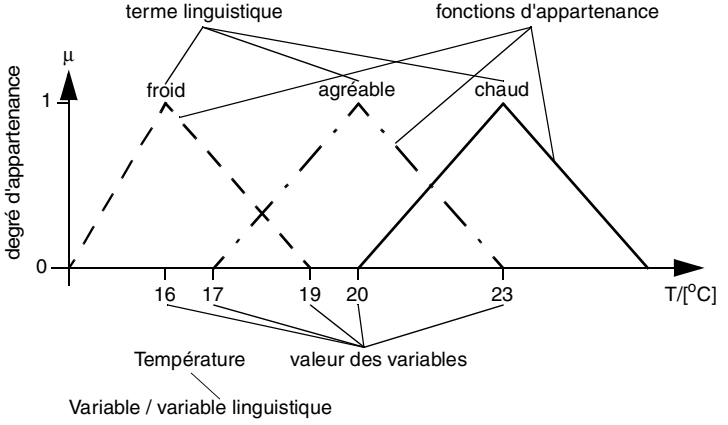
Exemple

Les termes linguistiques "froid", "agréable" et "chaud" ne se laissent pas clairement délimiter l'un par rapport à l'autre. Afin de mieux déterminer ces lignes de partage floue, chaque terme linguistique est évalué avec un degré d'appartenance.

Ce degré d'appartenancesignifie à présent, dans quelle mesure la température est vraiment froide.

- Un degré d'appartenance de 0 pour le teme linguistique "froid" signifie que la température n'est pas du tout froide.
- Un degré d'appartenance de 1 pour le teme linguistique "froid" signifie que la température est froide à 100% ; elle est vraiment froide. Les différentes notions sont à nouveau représentées en image.

Exemple pour un degré d'appartenance



Définition

Le degré d'appartenance définit dans quelle mesure une valeur physique est attribuée à un terme linguistique.

- 0 signifie que la valeur ne correspond pas du tout au terme linguistique.
- 1 signifie que la valeur correspond complètement au terme linguistique.

Fonction d'appartenance

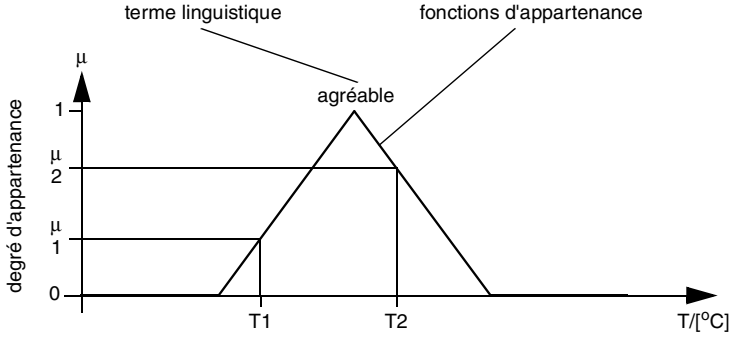
Exemple

La transition douce entre "pas vraiment froid" à "vraiment froid" est décrit par une fonction. De telles fonctions sont dénommées des fonctions d'appartenance, ensuite elles décrivent le degré d'appartenance de chaque valeur physique à son terme linguistique.

Pour la variable température, le degré d'appartenance ne décrit pas seulement si une température est "froide" ou non, la fonction d'appartenance attribue à chaque valeur physique une valeur de fiabilité avec l'expression "la température est froide", ceci est le degré d'appartenance.

En général, il suffit de déterminer les appartenances par des variations de la fonction relativement simples, comme les triangles, les trapèzes ou les rampes.

Un degré d'appartenance peut ainsi être déterminé par chaque valeur physique de la variable température.



(T1, μ1), (T2, μ2) : éléments de la quantité de Fuzzy

Dans cet exemple, le degré d'appartenance est déterminé pour deux valeur de température différente. Si l'on a attribué une fonction d'appartenance à un terme linguistique, on peut alors déterminer l'appartenance de la variable température au terme linguistique correspondant à l'aide du degré d'appartenance.

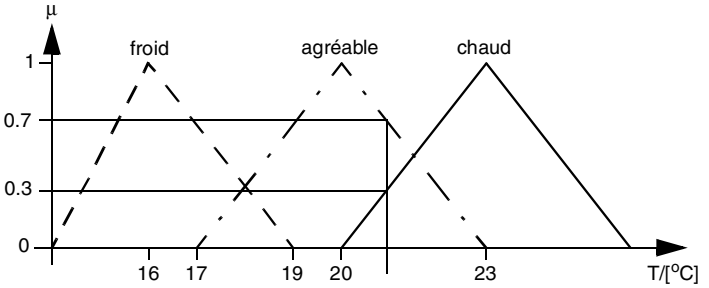
Définition

La fonction d'appartenance attribue pour chaque valeur physique un degré d'appartenance à un terme linguistique.

Fuzzification

Exemple

Fuzzification de la valeur d'entrée température



La valeur d'entrée dans la représentation traditionnelle "précise" par rapport à la représentation floue du Fuzzy

- Représentation précise : La température est de 21 degrés C.
- Représentation Fuzzy :
  La température est :
  - froide jusqu'au degré d'appartenance 0.0
  - agréable jusqu'au degré d'appartenance 0.7
  - chaude jusqu'au degré d'appartenance 0.3

Définition

On appelle fuzzification la détermination du degré d'appartenance pour tous les termes linguistiques d'une variable rapportée à une valeur d'entrée.

Règles

Exemple

Lorsque nous considérons nos activités quotidiennes, nous constatons que nos activités sont basées sur diverses hypothèses. Nous accomplissons une chose lorsqu'une hypothèse est rencontrée. Pour décrire l'hypothèse/réaction, nous utilisons des règles.

Par exemple, nous disons :

- Lorsque la température est ressentie comme étant froide, augmentons un peu le chauffage
  ou bien nous disons :
- Lorsque la température est ressentie comme étant chaude, diminuons un peu le chauffage.

Une autre règle pourrait être la suivante :

- Lorsqu'en été la température extérieure n'est pas froide, débranchons complètement le chauffage.

Derrière ces règles se cache le savoir d'un expert, qui, par une étude plus ou moins coûteuse, a acquis la faculté d'influer sur un processus dans un sens souhaité. Le savoir de l'expert ne se limite ici pas uniquement à la manière de procéder (augmentons un peu la température), mais est en même temps relié à la connaissance de ce qu'"un peu" signifie, par exemple, comment tourner le thermostat.

Définition

Une règle représente une déclaration (floue) sur la valeur de sortie pour une valeur d'entrée déterminée (floue) .

Opérateurs

Exemple

La plupart du temps, des conditions (prémisses) sont adjointes aux règles par les expressions orales "ET" et "OU" . Dans l'exemple, les prémisses "En été" (la variable de base est ici la saison et le terme linguistique est l'été) est associée à la prémisse "lorsque la température extérieure n'est pas froide" de manière telle que les deux déclarations doivent s'appliquer pour pour en tirer la conclusion.

Ces associations ET et OU sont traitées dans la technique Fuzzy par des opérateurs mathematiques. Dans la théorie Fuzzy, il y a une série d'opérateurs qui réalisent d'un côté l'association ET et d'un autre l'association OU. Le plus simple opérateur pour l'association ET est le "minimum" et le plus simple opérateur pour l'association OU est le "maximum".

Définition

L'association des conditions s'effectue par des opérateurs.

Pondération des règles

Lors de la fixation de règles, il peut se passer que celles-ci possèdent différentes valences. Ainsi, par exemple, une règle est toujours confirmée (à 100%), une autre pas toujours (à 80%). Pour pouvoir exprimer ceci, la possibilité existe d'attribuer à chaque règle un degré de vraisemblance. En général, ceci est atteint mathematiquement en multipliant le résultat de l'association d'une règle avec le degré de vraisemblance.

Inférence

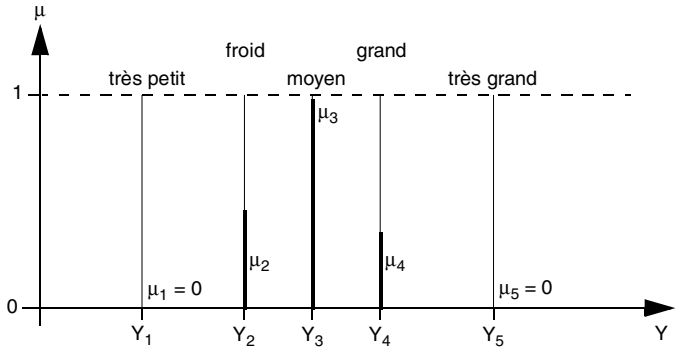
L'inférence est le résultat du traitement de toutes les règles

Défuzzification

Exemple

La façon la plus simple de transformer une variable floue Fuzzy en une variable précise consiste à ce que chaque attribut de la variable Fuzzy livre une proposition pour une variable précise. Chaque terme linguistique est ainsi représenté par une valeur numérique fixe pour la variable. On parle ici de singeltons. Les propositions faites pour chacun des termes linguistiques sont alors pondérés par les degrés d'appartenance correspondants. De cette manière, la variable Fuzzy devient une variable précise

Singletons



Note : Il y a également dans la théorie Fuzzy une série d'autres méthodes de défuzzification, qui se caractérisent partiellement par une très grande abondance de calculs. Celles-ci ne seront pas considérées plus en détail à cet endroit.

Définition

On appelle défuzzification, la transformation de variables floues (Fuzzy) en variables précises .

2.2 Fuzzy-Control dans Concept

Résumé

Introduction

Ce chapitre décrit comment la logique Fuzzy est configuée et quels EFB de ConCept doivent être à cet effet mis à disposition.

Contenu de ce sous-chapitre

Ce sous-chapitre contient les sujets suivants :

Les EFB dans la Bibliothèque Fuzzy

Les EFB dans la Bibliothèque Fuzzy

Les EFB pour INTet REALArithmétique

Opération	EFB	Groupe	Description
Fuzzification	FUZ_STERM_INT (voir *FUZ_STERM_*** : Fuzzification d'un terme, p. 67*), FUZ_STERM_REAL (voir *FUZ_STERM_*** : Fuzzification d'un terme, p. 67*)	Fuzzyfy	Fuzzification d'un terme
Fuzzification	FUZ_ATERM_INT (voir *FUZ_ATERM_ INT, FUZ_ATERM_REAL : Fuzzification de tous les termes, p. 49*), FUZ_ATERM_REAL (voir *FUZ_ATERM_ INT, FUZ_ATERM_REAL : Fuzzification de tous les termes, p. 49*)	Fuzzyfy	Fuzzification de jusqu'à 9 termes en une fois
Fuzzification	FUZ_ATERM_STI (voir *FUZ_ATERM_STI, FUZ_ATERM_STR : Fuzzification de tous les termes (structure), p. 55*), FUZ_ATERM_STR (voir *FUZ_ATERM_STI, FUZ_ATERM_STR : Fuzzification de tous les termes (structure), p. 55*)	Fuzzyfy_Struct	Fuzzification de jusqu'à 9 termes en une fois. classé le résultat dans la structure de données.
Inférence	FUZ_MAX_INT (voir *FUZ_MAX_*** : Fuzzy Maximum, p. 59*), FUZ_MAX_REAL (voir *FUZ_MAX_*** : Fuzzy Maximum, p. 59*)	Operators_OR	Opérateurs OU : Maximum
Inférence	FUZ_MIN_INT (voir *FUZ_MIN_*** : Fuzzy Minimum, p. 61*), FUZ_MIN_REAL (voir *FUZ_MIN_*** : Fuzzy Minimum, p. 61*)	Operators_AND	Opérateurs ET : Minimum

Opération	EFB	Groupe	Description
Inférence	FUZ_SUM_INT (voir *FUZ_SUM_*** : somme Fuzzy, p. 75*), FUZ_SUM_REAL (voir *FUZ_SUM_*** : somme Fuzzy, p. 75*)	Operators_OR	Opérateurs OU : Somme
Inférence	FUZ_PROD_INT (voir *FUZ_PROD_*** : produit Fuzzy , p. 63*), FUZ_PROD_REAL (voir *FUZ_PROD_*** : produit Fuzzy , p. 63*)	Operators_AND	Opérateurs ET : Produit
défuzzification	DEFUZ_INT (voir *DEFUZ_INT, DEFUZ_REAL : Défuzzification avec singletons, p. 39*), DEFUZ_REAL (voir *DEFUZ_INT, DEFUZ_REAL : Défuzzification avec singletons, p. 39*)	Defuzzify	Défuzzifier avec des singletons
défuzzification	DEFUZ_STI (voir *DEFUZ_STI, DEFUZ_STR : Défuzzification avec singletons (structure), p. 45*), DEFUZ_STR (voir *DEFUZ_STI, DEFUZ_STR : Défuzzification avec singletons (structure), p. 45*)	Defuzzify_Struct	Défuzzifier avec des singletons. retirer les entrées de la structure de données.

La différence entre l'arithmétique des entiers et l'arithmétique des réels est :

- la résolution à l'intérieur du calcul.
  La résolution du degré d'appartenance est de 0.01 %.
  Le domaine du degré d'appartenance 0...1 est cadré à 0...10 000.
- la possibilité de travailler dans l'arithmétique des réels avec des valeurs physiques.
- le temps d'exécution qui est plus grand que dans l'arithmétique des réels.

Fuzzification

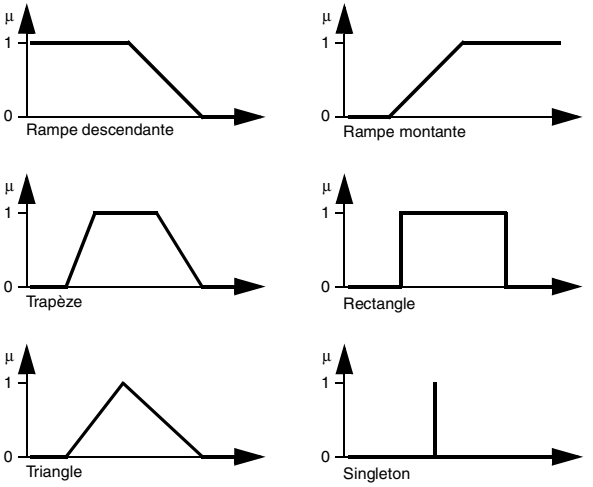
Introduction

Toutes les variables, qui doivent être associées au moyen de règles linguistiques, sont fuzzifiées. Pour cela, il faut d'abord évaluer combien d'attributs d'une variable doivent être associés. Ceci dépend de de la nécessité de l'utilisation de ces attributs dans les règles. La plupart du temps, de 3 à 5 de ces attributs suffisent.Avec ConCept, vous avez la possibilité de choisir une fuzzification pour chaque terme d'une variable ou une fuzzification de jusqu'à 9 termes d'une variable.

Fuzzification avec FUZ_STERM

La fuzzification d'un terme isolé s'effectue avec la fonction d'appartenance, qui soutient jusqu'à 4 points d'appui.

La fonction d'appartenancen peut p.ex. prendre les formes suivantes :

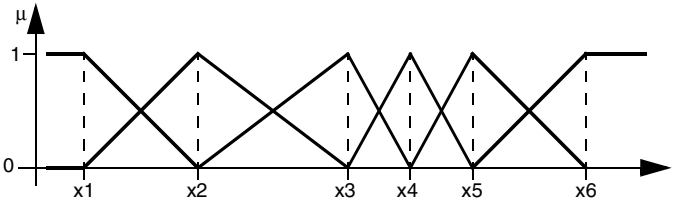


Vous trouverez de plus amples informations à ce sujet dans la description du EFB FUZ_STERM (voir *FUZ_STERM_*** : Fuzzification d'un terme, p. 67*).

Fuzzification avec FUZ_ATERM ou FUZ_ATERM_ST

En général, les fonctions les plus souvent employées sont la rampe et le triangle. La fuzzification d'un terme isolé offre une grande flexibilité pour des utilisateurs expérimentés du Fuzzy, mais la manière la plus simple de fuzzifié une variable est d'utiliser l'EFB FUZ_ATERM (voir *FUZ_ATERM_ INT, FUZ_ATERM_REAL : Fuzzification de tous les termes, p. 49*) ou FUZ_ATERM_ST (voir *FUZ_ATERM_STI, FUZ_ATERM_STR : Fuzzification de tous les termes (structure), p. 55*) comme ces EFB ne fuzzifient non seulement un terme, mais tous les termes d'une variable Fuzzy. Ces EFB simplifient également l'entrée de la fonction d'appartenance pour différents termes.

L'idée qui soutend cette simplification est que les différentes fonctions d'appartenance de tous les termes d'une variable Fuzzy sont dépendants l'un de l'autre. Dans la plupart des cas, la somme du degré d'appartenance de deux fonctions qui se suivent 1. Si, pour une plus grande simplification, on n'utilise que des fonctions triangle au lieu de la fonction trapèze, il s'ensuit l'exemple suivant de la définition d'une fonction d'appartenance pour 6 termes



Dans ce cas, il ne faut définir que 6 points de base, au lieu de 2 x 2 + 4 x 3 = 14 points de base dans le cas d'i,e fuzzification avec FUZ_STERM (voir *FUZ_STERM_*** : Fuzzification d'un terme, p. 67*). L'EFB FUZ_ATERM (voir *FUZ_ATERM_ INT, FUZ_ATERM_REAL : Fuzzification de tous les termes, p. 49*) / FUZ_ATERM_ST (voir *FUZ_ATERM_STI, FUZ_ATERM_STR : Fuzzification de tous les termes (structure), p. 55*) permet un maximum de 9 attributs.

La fuzzification doit être exécutée pour chaque variable d'entrée. Le résultat de chaque fuzzification est des degrés d'appartenance pour chaque terme qui est associé aux attributs. Lors de l'utilisation de l'EFB FUZ_ATERM (voir *FUZ_ATERM_ INT, FUZ_ATERM_REAL : Fuzzification de tous les termes, p. 49*), le résultat est l'élaboration de chaque variable pour l'utilisation de liaisons grafiques dans l'inférence. Lors de l'utilisation de l'EFB FUZ_ATERM_ST (voir *FUZ_ATERM_STI, FUZ_ATERM_STR : Fuzzification de tous les termes (structure), p. 55*), les degrés d'appartenance sont résumés dans une structure de données.

Inférence

Conversion des règles

Les règles sont converties au moyen d'opérateurs par la liaison du degré d'appartenance de la variable Fuzzy. Les entrées des EFB, qui représentent les opérateurs Fuzzy, sont associées aux degrés d'appartenance (produit par les EFB-Fuzzy). Les paires FUZ_MIN (voir *FUZ_MIN_*** : Fuzzy Minimum, p. 61*)/ FUZ_MAX (voir *FUZ_MAX_*** : Fuzzy Maximum, p. 59*) et FUZ_PROD (voir *FUZ_PROD_*** : produit Fuzzy , p. 63*)/FUZ_SUM (voir *FUZ_SUM_*** : somme Fuzzy, p. 75*) se sont révélées être de bonnes combinaisons pour les associations ET/OU. Pour une première tentative, utilisez la paire FUZ_MIN (voir *FUZ_MIN_*** : Fuzzy Minimum, p. 61*)/FUZ_MAX (voir *FUZ_MAX_*** : Fuzzy Maximum, p. 59*).

Pondération des règles

Les règles doivent être pondérées par une multiplication. Pour l'arithmetique des réels, on se sert du EFB FUZ_PROD_REAL (voir *FUZ_PROD_*** : produit Fuzzy , p. 63*). Pour l'arithmetique des entiers, on se sert de l'EFB FUZ_PROD_INT (voir *FUZ_PROD_*** : produit Fuzzy , p. 63*), qui considère la forme normalisée du degré d'appartenance de 0 ... 10 000.

Défuzzification

Principe

Les résultats de l'utilisation de toutes les règles sont de nouveau des degrés d'appartenance aux attributs de la variable de sortie. Pour atteindre un résultat exploitable (des attributs avec des degrés d'appartenance ne peuvent pas être utilisé directement p.ex. pour la commande de valeurs), les degrés d'appartenance de tous les termes des variables doivent être combinés de manière sensée. Ceci peut être obtenu par le bloc fonction de défuzzification DEFUZ (voir *DEFUZ_INT, DEFUZ_REAL : Défuzzification avec singletons, p. 39*).

Manière de travailler dans Concept

Le bloc fonction DEFUZ (voir *DEFUZ_INT, DEFUZ_REAL : Défuzzification avec singletons, p. 39*) produit au moyen de singleton à partir des degrés d'appartenance une valeur univoque pour les variables de sortie, qui sont attribuées aux termes des variables de sortie.

Exemple pour Concept

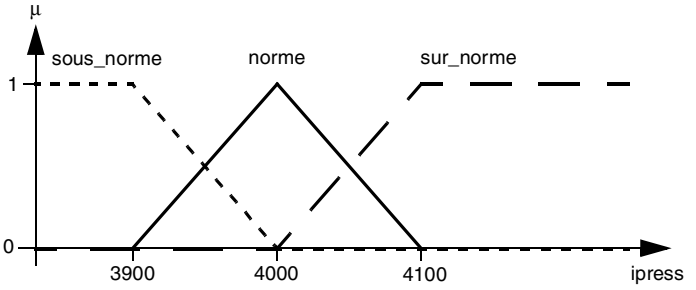
Handicaps

L'exemple est basé sur la logique des nombres entiers. Il y a 2 variables, ipres und itemp, qui, d'après quatre règles prédéfinies doivent être évaluées.

Fuzzification de la variable "ipress" (ipress a 3 termes linguistiques)

- sous_norme
- norme
- sur_norme

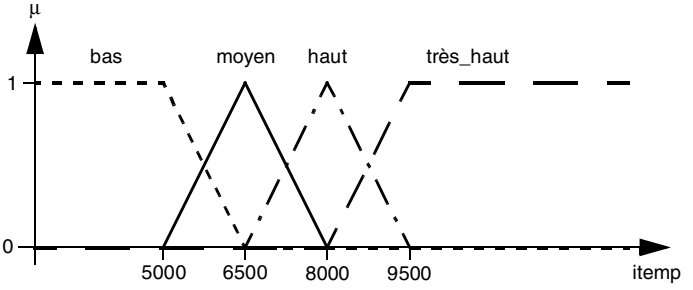
Fuzzification de la variable "ipress"



Fuzzification de la variable "itemp" (itemp a 4 termes linguistiques)

- bas
- moyen
- haut
- très_haut

Fuzzification de la variable "itemp"



Sont valables les quatres règles suivantes :

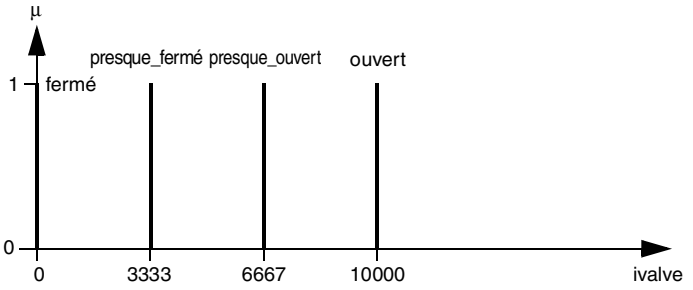
- IF ipress = norme AND itemp = haut THEN valve= presque_fermé
- IF ipress = sur_norme AND itemp = très_haut THEN valve = fermé
- IF ipress = norme AND itemp = moyen THEN valve= presque_ouvert
- IF ipress = sous_norme AND itemp = bas THEN valve = ouvert

La défuzzification doit s'effectuer avec des singletons

défuzzification de la variable "ivalve" ((ivalve a 4 termes linguistiques)

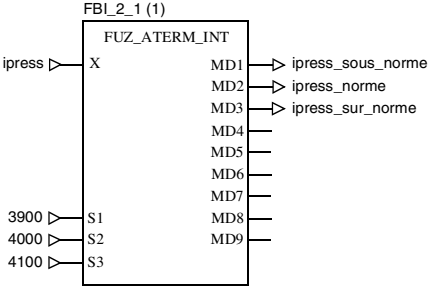
- fermé
- presque_fermé
- presque_ouvert
- ouvert

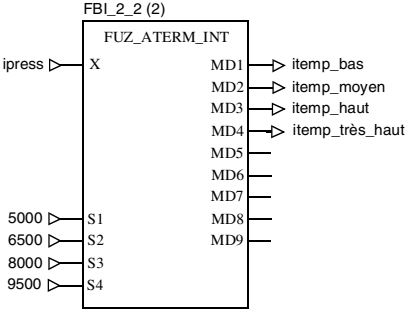
défuzzification de la variable "ivalve"



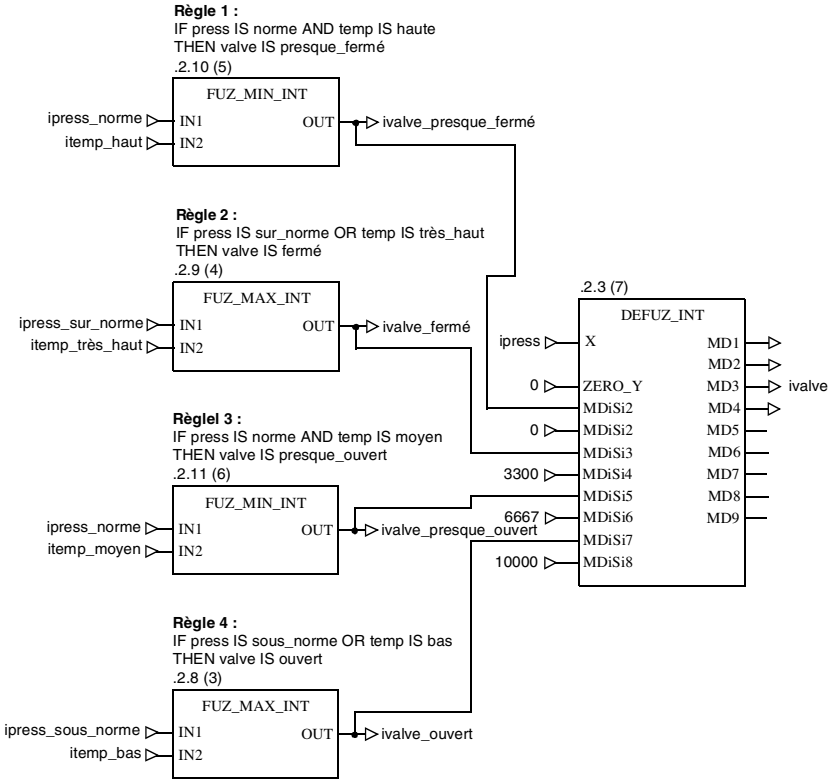
Conversion dans Concept

Fuzzification :





DéFuzzification :



Descriptions de l'EFB

II

Résumé

Introduction

Ces descriptions de l'EFB sont rangées dans l'ordre alphabétique.

Note : Le nombre des entrées de certains EFB peut être augmenté à max. 32 par une élévation verticale des symboles FFB. Quels que soient les EFB concernés, prélevez la description de chaque EFB.

Contenu de cette partie

Cette partie contient les chapitres suivants :

DEFUZ_INT, DEFUZ_REAL : Défuzzification avec singletons

3

Résumé

Introduction

Ce chapitre décrit les blocs :

- DEFUZ_INT
- DEFUZ_REAL

Contenu de ce chapitre

Ce chapitre contient les sujets suivants :

Présentation

Présentation

La fonction defuzzifie des termes linguistiques, qui sont représentés par des singletons, suivant la méthode de la moyenne maximum. La position des singletons est définie par les points d'appui (S1 ... S9). Chaque terme est pondéré par son degré d'appartenance propre (MD1 ... MD9) . La gamme de valeur du degré d'appartenance s'élève pour le type de données INT de 0 ... à 10 000 et pour le type de données REAL 0 ... 1. La signification des entrées (entrées extensibles MDiSi) doit être trouvée dans la description du paramètre*Description des paramètres, p. 41*.

Le nombre des entrées (MDiSi) peut être augmenté jusqu'à 18 en modifiant verticalement la taille des modules. Ceci correspond à 9 singletons. Il n'est pas possible de configurer d'autres entrées.

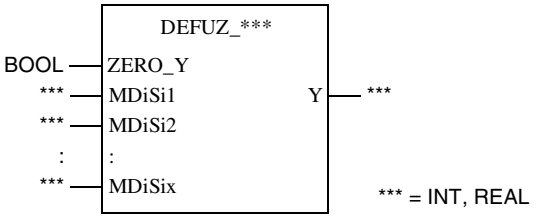
Les types de données de toutes les valeurs d'entrées (MDiSi) et celui de la valeur de sortie doivent être identiques. Un bloc fonction propre est à chaque fois disponible pour l'élaboration des différents types de données.

Comme paramètres supplémentaires, on peut configurer EN et ENO.

Représentation

Symbole

Représentation du bloc :



Formule

Formule du bloc :

$$Y = \frac{\sum_{i=1}^{n} MDi \times Si}{\sum_{i=1}^{n} MDi}$$

Explication : n = nombre de singletons

Condition : $2 \leq n \leq 9$

Description des paramètres

Description des paramètres du bloc :

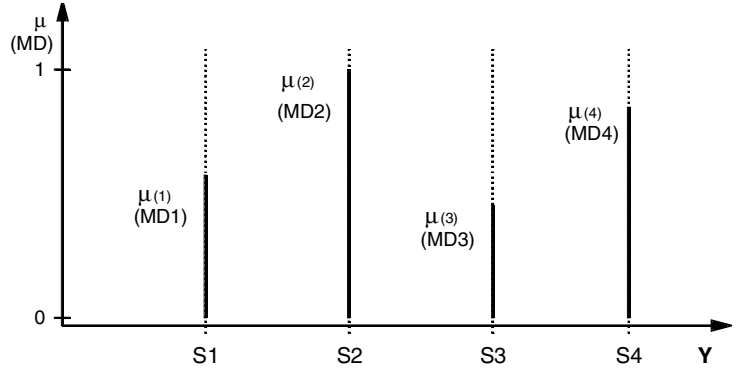
Paramètres	Types de données	Signification
ZERO_Y	BOOL	0 : Sortie de la dernière valeur Y
MDiSi2	INT, REAL	Singleton : 1; MD1 degré d'appartenance (Membership Degree)
MDiSi2	INT, REAL	Singleton : 1; S1 point d'appui
MDiSi3	INT, REAL	Singleton : 2; MD2 degré d'appartenance (Membership Degree)
MDiSi4	INT, REAL	Singleton : 2; S2 point d'appui
:	:	:
MDiSi17	INT, REAL	Singleton : 9; MD9 degré d'appartenance (Membership Degree)
MDiSi18	INT, REAL	Singleton : 9; S9 point d'appui
Y	INT, REAL	Sortie

Description détaillée

Description de la fonction

Pour la défuzzification de variables linguistiques, on dispose d'un bloc fonction DEFUZ pour la méthode de la moyenne maximum (Mean of Maximum Method) avec singletons comme fonction d'appartenance.

La position des singletons est définie par des points d'appui.



Lors de la configuration des blocs fonction, veillez à ce que les entrées soient toujours utilisés par paires, puisque chaque valeur linguistique est pondérée par son dégré d'appartenance. La signification de l'indice des entrées est représentée dans la description des paramètres *Description des paramètres, p. 41* .

La sortie Y est placée sur 0.

Si tous les degrés d'appartenance ont la valeur 0, le comportement du bloc fonction peut être déterminé par la sortie ZERO_Y :

valeur ZERO_Y	**résultat**
ZERO_Y = 0	La sortie Y reste inchangée.
ZERO_Y = 1	La sortie Y est placée sur 0.

Erreur due au temps de transit

Erreur due au temps de transit

Un message d'erreur survient , lorsque

- plus de 9 points d'appui (ceci correspond à max. 18 entrées MDiSi) ont été projetées (E_EFB_TOO_MANY_INPUTS),
- le bloc fonction a été paramétré avec un nombre d'entrées impair (E_EFB_WRONG_NUMBER_OF_INPUTS) ou
- si un des degrés d'appartenance MD1 ... MD9 est en dehors de la gamme de valeurs (E_EFB_INPUT_VALUE_OUT_OF_RANGE). Seules les plages de valeur suivantes sont possibles :
  - INT : 0 ... 10 000
  - REAL : 0 ... 1

DEFUZ_STI, DEFUZ_STR : Défuzzification avec singletons (structure)

4

Résumé

Introduction

Ce chapitre décrit les blocs :
- DEFUZ_STI
- DEFUZ_STR

Contenu de ce chapitre

Ce chapitre contient les sujets suivants :

Présentation

Description de la fonction

La fonction défuzzifie des termes linguistiques, qui sont représentés par des singletons, suivant la méthode de la moyenne maximum. La position des singletons est définie par les points d'appui (S1 ... S9). Chaque valeur linguistique est pondérée par son degré d'appartenance (term1 à term9) à la structure de données FUZ_MD_INT (pour DEFUZ_STI) ou FUZ_MD_REAL (pour DEFUZ_STR). La gamme de valeur du degré d'appartenance compred pour le type de données INT 0 ... 10 000 et pour le type de données REAL 0 ... 1.

Le nombre d'entrées (S1 ... Sn) peut être élevé à 9 max. par une modification verticale du cadre du bloc. Il n'est pas possible de configurer d'autres entrées.

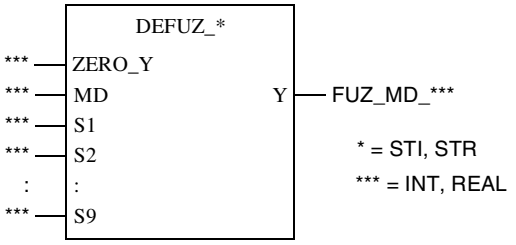
Les types de données de toutes les valeurs d'entrée (Sn) doivent être identiques. Un bloc fonction propre est à chaque fois disponible pour l'élaboration des différents types de données.

Comme paramètres supplémentaires, on peut configurer EN et ENO.

Représentation

Symbole

Représentation du bloc :



Formule

Formule du bloc

$$Y = \frac{\sum_{i=1}^{n} MDi \times Si}{\sum_{i=1}^{n} MDi}$$

Explication : n = nombre de singletons

Condition : $2 \leq n \leq 9$

Description des paramètres

DEFUZ_STI, DEFUZ_STR

Paramètres	Type de données	Signification
ZERO_Y	BOOL	0: Sortie de la dernière valeur Y 1: la sortie Y est mise à "0"
MD	FUZ_MD_INT, FUZ_MD_REAL	Degré d'appartenance (Membership Degree) (term1 ... term9)
S1	INT, REAL	Singleton : 1
S2	INT, REAL	Singleton : 2
:	:	:
S9	INT, REAL	Singleton : 9
Y	INT, REAL	Sortie

FUZ_MD_INT, FUZ_MD_REAL

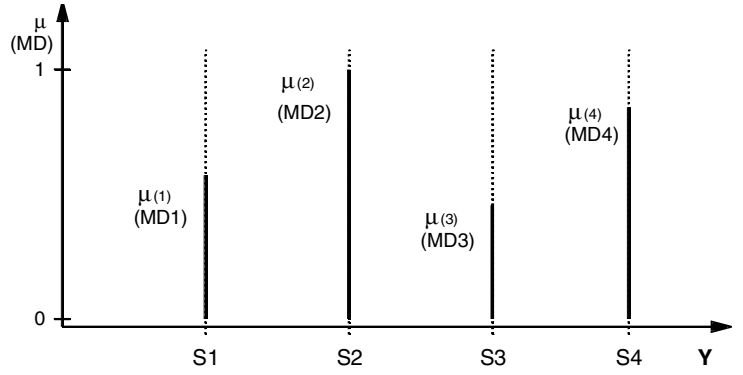
Elément	Type de données	Signification
n	INT	nombre de termes
term1	INT, REAL	Degré d'appartenance (Membership Degree) (MD1)
:	:	:
term9	INT, REAL	Degré d'appartenance (Membership Degree) (MD9)

Description détaillée

Description de la fonction

Pour la défuzzification de variables linguistiques, on dispose de blocs de fonction DEFUZ_ST pour la méthode de la moyenne maximum (Mean of Maximum Method) avec singletons comme fonction d'appartenance.

La position des singletons est définie avec les points d'appui (S1 ... S9).



Chaque valeur linguistique est pondérée par son degré d'appartenance.

Si tous les degrés d'appartenance ont la valeur 0, le comportement du bloc fonction peut être déterminé par la sortie ZERO_Y :

valeur ZERO_Y	**résultat**
ZERO_Y = 0	La sortie Y reste inchangée.
ZERO_Y = 1	La sortie Y est placée sur 0.

Erreur due au temps de transit

Erreur due au temps de transit

Un message d'erreur (E_EFB_TOO_MANY_INPUTS) survient, lorsque plus de 9 points d'appui ont été configurées.

Un avertissement (E_EFB_WRONG_NUMBER_OF_INPUTS) survient, lorsque le nombre d'entrées (= nombres des points d'appui S1 ... Sn) ne correspond pas avec le nombre des termes employés (MD) dans la structure de données FUZ_MD_INT.n (lors de DEFUZ_STI) ou FUZ_MD_REAL.n (lors de DEFUZ_STR). Dans ce cas, le calcul de la valeur de sortie n'intègre que le nombre de points d'appui correspondant au nombre de termes utilisés (MD) dans la structure de données FUZ_MD_INT.n (pour DEFUZ_STI) ou FUZ_MD_REAL.n (pour DEFUZ_STR).

FUZ_ATERM_ INT, FUZ_ATERM_REAL : Fuzzification de tous les termes

5

Aperçu

Introduction

Ce chapitre décrit les blocs :
- FUZ_ATERM_INT
- FUZ_ATERM_REAL

Contenu de ce chapitre

Ce chapitre contient les sujets suivants :

Présentation

Description de la fonction

Le bloc fonction fuzzifie jusqu'à 9 termes des variables linguistiques (entrée X) et indique chaque degré d'appartenance (sorties MD1 ... MD9). La gamme de valeur à la sortie pour le type de données INT comprend 0 ... 10 000 et pour le type de données REAL 0 ... 1. Les fonctions d'appartenance sont définies par les points d'appui (entrées extensibles S1 ... S9).

Le bloc fonction travaille avec une simplification propre à la définition des fonctions d'appartenance :

- rampes pour la première et la dernière fonction d'appartenance
- triangles pour les fonctions d'appartenance entre la première et la dernière
- la somme de deux degrés d'appartenance de deux termes linguistiques successifs est toujours 1 (10 000)
- la somme des degrés d'appartenance de tous les termes linguistiques pour chaque valeur d'entrée X est toujours 1 (10 000)

Le nombres des points d'appui (S1 ... Sx) peut être élevé à 9 max. par un agrandissement vertical du cadre de bloc. Il n'est pas possible de configurer d'autres points d'appui.

Le nombre de degrés d'appartenance calculés correspond au nombre de fonctions d'appartenance. Si la configuration comprend moins de neuf fonctions d'appartenance, les sorties restantes adoptent la valeur 0. (p.ex. pour 4 fonctions d'appartenance, on compte 4 degrés d'appartenance pour MD1 ... MD4 et MD5 ... MD9 sont à 0).

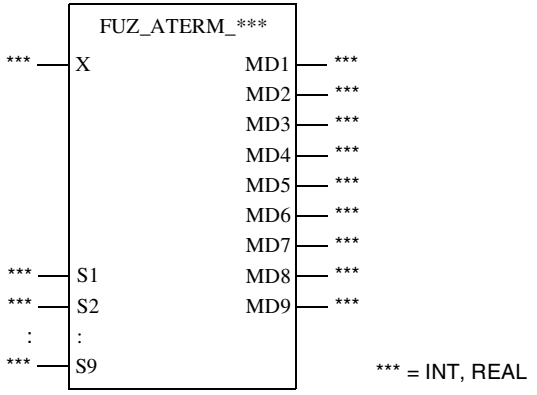
Les types de données de toutes les valeurs d'entrée et celles des valeurs de sortie doivent être identiques. Un bloc fonction propre est à chaque fois disponible pour l'élaboration des différents types de données.

Comme paramètres supplémentaires, on peut configurer EN et ENO.

Représentation

Symbole

Représentation du bloc :



Description des paramètres

Description des paramètres du bloc :

Paramètres	Type de données	Signification
X	INT, REAL	Variable linguistique
S1	INT, REAL	Point d'appui S1
S2	INT, REAL	Point d'appui S2
:	:	:
S9	INT, REAL	Point d'appui S9
MD1	INT, REAL	Sortie degré d'appartenance MD1 (Membership Degree)
MD2	INT, REAL	Sortie degré d'appartenance MD2 (Membership Degree)
:	:	:
MD9	INT, REAL	Sortie degré d'appartenance MD9 (Membership Degree)

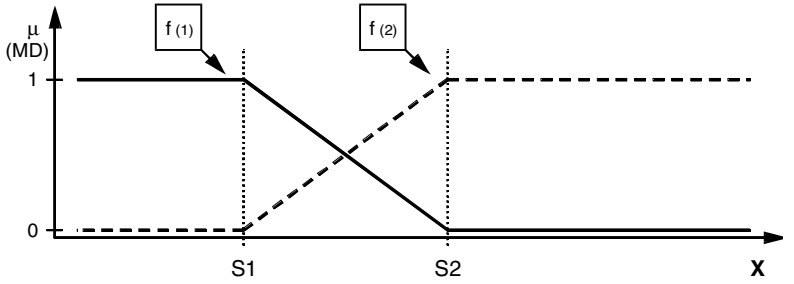
Description détaillée

Description des paramètres

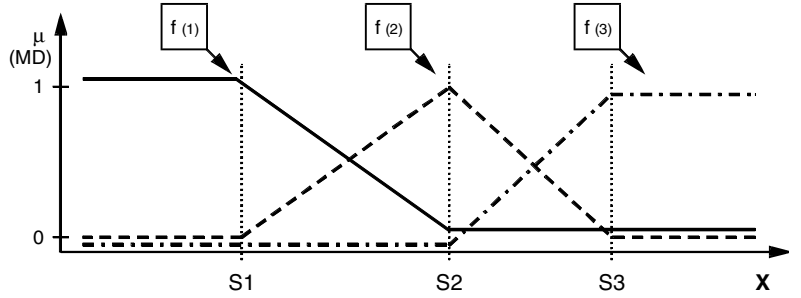
Avec le bloc fonction FUZ_ATERM, tous les termes d'une variable linguistique peuvent être fuzzifiés en même temps. Les fonctions d'appartenance sont déterminées par des points d'appui (S1, S2, S3,...). Le concept de cette fuzzification permet de définir les extrémités de plusieurs fonctions d'appartenance avec un point d'appui à la fois. Ces fonctions d'appartenance se présentent sous forme de rampes et de triangles, la somme des différents degrés d'appartenance étant toujours 100 %. Ces corrélations sont présentées aux diagrammes temporels suivants.

Diagramme de cycles

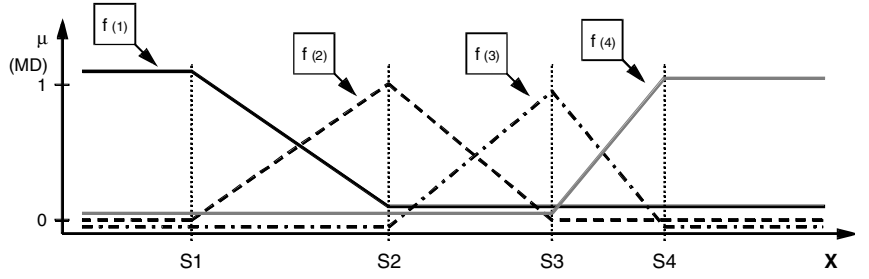
2 fonctions d'appartenance, 2 points d'appui, 2 degrés d'appartenance



3 fonctions d'appartenance, 3 points d'appui, 3 degrés d'appartenance



4 fonctions d'appartenance, 4 points d'appui, 4 degrés d'appartenance



Erreur due au temps de transit

Erreur due au temps de transit

Un message d'erreur survient , lorsque

- les points d'appui ne sont pas rangées dans l'ordre (E_EFB_INPUT_VALUE_OUT_OF_RANGE). Les points d'appui doivent être rangées dans l'ordre croissant (S1 < S2 < S3 < ... < S9) ou
- la configuration comporte plus de 9 points d'appui (E_EFB_TOO_MANY_INPUTS).

FUZ_ATERM_STI, FUZ_ATERM_STR : Fuzzification de tous les termes (structure)

6

Résumé

Introduction

Ce chapitre décrit les blocs :
- FUZ_ATERM_STI
- FUZ_ATERM_STR

Contenu de ce chapitre

Ce chapitre contient les sujets suivants :

Présentation

Description de la fonction

La fonction fuzzifie jusqu'à 9 termes des variables linguistiques (entrée X) et sort en output chacun des degrés d'appartenance dans la structure de donnéesFUZ_MD_INT (dans FUZ_ATERM_STI) ou FUZ_MD_REAL (dans FUZ_ATERM_STR) dans les éléments term1 ... term9. La gamme de valeur à la sortie pour le type de données INT comprend 0 ... 10 000 et pour le type de données REAL 0 ... 1. Les fonctions d'appartenance sont définies par les points d'appui (entrées extensibles S1 ... S9).

Le bloc fonction travaille avec une simplification propre à la définition de la fonction d'appartenance.

- rampes pour la première et la dernière fonction d'appartenance
- triangles pour les fonctions d'appartenance entre la première et la dernière
- la somme de deux degrés d'appartenance de deux termes linguistiques successifs est toujours 1 (10 000)
- la somme des degrés d'appartenance de tous les termes linguistiques pour chaque valeur d'entrée X est toujours 1 (10 000)

Le nombres des points d'appui (S1 ... Sx) peut être élevé à 9 max. par un agrandissement vertical du cadre de bloc. Il n'est pas possible de configurer d'autres points d'appui.

Le nombre des degrés d'appartenance calculés correspond au nombre de fonctions d'appartenance et classé dans la structure de données (Element n). Si vous configurez moins de neuf fonctions d'appartenance, les éléments restants de la structure de données (termx) ne sont pas modifiés. exemple : 4 points d'appui sont associés à 4 fonctions d'appartenance dont les 4 degrés d'appartenance sont appliqués aux éléments term1 à term4; term5 à term9 ne sont pas modifiés.) Ceci présente l'avantage que les structures de données constituées en mémoire d'état n'occupent que la place en mémoire dont elles ont vraiment besoin.

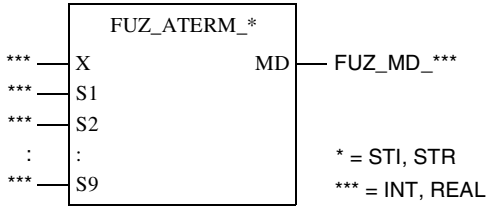
Les types de données de toutes les valeurs d'entrées doivent être identiques. Un bloc fonction propre est à chaque fois disponible pour l'élaboration des différents types de données.

Comme paramètres supplémentaires, on peut configurer EN et ENO.

Représentation

Symbole

Représentation du bloc :



Description des paramètres

FUZ_ATERM_STI, FUZ_ATERM_STR

Paramètre	Types de données	Signification
X	INT, REAL	Variable linguistique
S1	INT, REAL	Point d'appui S1
S2	INT, REAL	Point d'appui S2
:	:	:
S9	INT, REAL	Point d'appui S9
MD	FUZ_MD_INT, FUZ_MD_REAL	Sortie de degré d'appartenance (Membership Degree) (term1 ... term9)

FUZ_MD_INT, FUZ_MD_REAL

Elément	Types de données	Signification
n	INT	nombre de termes
term1	INT, REAL	Degré d'appartenance (Membership Degree) (MD1)
:	:	:
term9	INT, REAL	Degré d'appartenance (Membership Degree) (MD9)

Description détaillée

Description des paramètres et diagramme de cycle

Vous trouverez la description des paramètres et les diagrammes de cycle de ce bloc dans *Description détaillée, p. 52*

Erreur due au temps de transit

Erreur due au temps de transit

Un message d'erreur survient , lorsque

- les points d'appui ne sont pas rangées dans l'ordre (E_EFB_INPUT_VALUE_OUT_OF_RANGE). Les points d'appui doivent être rangées dans l'ordre croissant (S1 < S2 < S3 < ... < S9) ou
- la configuration comporte plus de 9 points d'appui (E_EFB_TOO_MANY_INPUTS).

FUZ_MAX_*** : Fuzzy Maximum

7

Résumé

Introduction

Ce chapitre décrit les blocs FUZ_MAX_***.

Contenu de ce chapitre

Ce chapitre contient les sujets suivants :

Présentation

Description de la fonction

La fonction reconnaît la plus grande valeur d'entrées et la sort en output à la sortie.

Les types de données INT et REAL peuvent être travaillées. La gamme de valeur des entrées et de la sortie comprend pour le type de données INT 0 ... 10 000 et pour le type REAL 0 ... 1.

Les types de données de toutes les valeurs d'entrée et celles des valeurs de sortie doivent être identiques. On dispose chaque fois d'une fonction propre pour l'élaboration des différents types de données.

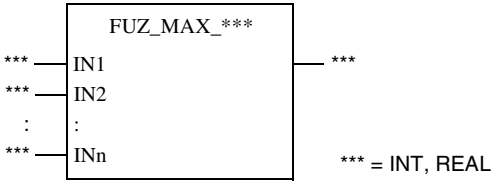
Le nombre des entrées peut être augmenté.

Comme paramètres supplémentaires, on peut configurer EN et ENO.

Représentation

Symbole

Représentation du bloc :



Formule

Formule du bloc :

$OUT = MAX\{0,IN1,IN2,\ldots,INn\}$

Description des paramètres

Description des paramètres du bloc :

Paramètre	Types de données	Signification
IN1	INT, REAL	1. Valeur d'entrée
IN2	INT, REAL	2. Valeur d'entrée
:	:	:
INn	INT, REAL	n. valeur d'entrée
OUT	INT, REAL	Valeur maximale

FUZ_MIN_*** : Fuzzy Minimum

8

Résumé

Introduction

Ce chapitre décrit les blocs FUZ_MIN_***.

Contenu de ce chapitre

Ce chapitre contient les sujets suivants :

Présentation

Description de la fonction

La fonction reconnaît la plus petite valeur d'entrée et la sort en output à la sortie.

Les types de données INT et REAL peuvent être travaillées. La gamme de valeur des entrées et de la sortie comprend pour le type de données INT 0 ... 10 000 et pour le type REAL 0 ... 1.

Les types de données de toutes les valeurs d'entrée et celles des valeurs de sortie doivent être identiques. On dispose chaque fois d'une fonction propre pour l'élaboration des différents types de données.

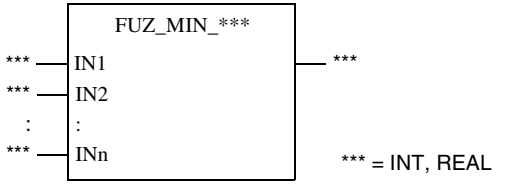
Le nombre des entrées peut être augmenté.

Comme paramètres supplémentaires, on peut configurer EN et ENO.

Représentation

Symbole

Représentation du bloc :



Formule

Formule du bloc :

REAL: OUT = MIN{1,IN1,IN2,…,INn}
INT: OUT = MIN{10000,IN1,IN2,…,INn}

Description des paramètres

Description des paramètres du bloc :

Paramètre	Types de données	Signification
IN1	INT, REAL	1. Valeur d'entrée
IN2	INT, REAL	2. Valeur d'entrée
:	:	:
INn	INT, REAL	n. valeur d'entrée
OUT	INT, REAL	Valeur maximale

FUZ_PROD_*** : produit Fuzzy

9

Résumé

Introduction

Ce chapitre décrit les blocs FUZ_PROD_***.

Contenu de ce chapitre

Ce chapitre contient les sujets suivants :

Présentation

Description de la fonction

La fonction forme le produit (sortie MD) des degré d'appartenances (entrées extensibles MD1 ... MDx). En outre, la fonction effectue (pour le calcul des entiers) une multiplication en tenant compte de la gamme de valeur du degré d'appartenance (0 ... 10 000). Les gammes de valeur aux entrées et à la sortie comprend pour le type de données INT 0 ... 10 000 et pour le type de données REAL 0 ... 1.

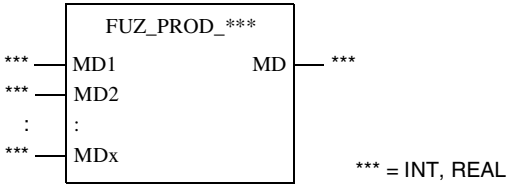
Le nombre des entrées peut être augmenté.

Comme paramètres supplémentaires, on peut configurer EN et ENO.

Représentation

Symbole

Représentation du bloc :



Description des paramètres

description des paramètres du bloc :

Paramètre	Types de données	Signification
MD1	INT, REAL	1. degré d'appartenance (Membership Degree)
MD2	INT, REAL	2. degré d'appartenance (Membership Degree)
:	:	:
MDx	INT, REAL	x. degré d'appartenance (Membership Degree)
MD	INT, REAL	Sortie produit (fuzzy)

Description détaillée

Description de la fonction

Dans l'arithmétique Real, on atteint la formation du produits du degré d'appartenance par une simple multiplication.

Règle	Exemple
(0 ... 1) * (0 ... 1) = (0 ... 1)	0.3 * 0.6 = 0.18

Dans l'arithmetique des entiers , une correction doit être effectuée, conditionnée par une remise à échelle de la gamme des valeurs :

Règle	Exemple
(0 ... 10 000) * (0 ... 10 000) = (0 ... 10 000)	3 000 * 6 000 = 1 800 (!)

FUZ_STERM_*** : Fuzzification d'un terme

10

Résumé

Introduction

Ce chapitre décrit les blocs FUZ_STERM_***.

Contenu de ce chapitre

Ce chapitre contient les sujets suivants :

Présentation

Description de la fonction

La fonction fuzzifie un seul terme de la variable linguistique (entrée X) et donne son degré d'appartenance à la sortie MD. La gamme de valeur de la sortie comprend pour le type de données INT 0 ... 10 000 et pour le type de données REAL 0 ... 1. La fonction d'appartenance est établie au moyen de 4 points d'appui maximum (S1 ... S4). Vous avez le choix entre 2 à 4 entrées (points d'appui). Pour 2 points d'appui, la fonction réalise une fonction rampe. Pour 3 points d'appui, on trouve une fonction triangle et pour 4 points d'appui une fonction trapèze. En paramétrant les points d'appui de manière différente, plusieurs courbes de fonction peuvent être réalisées.

Note : En paramétrant avec 4 points d'appui, le traitement de la fonction d'appartenance correspond au SFB 361 FUZZYFY du AKF125.

Les types de données de toutes les valeurs d'entrée et celles des valeurs de sortie doivent être identiques. Un bloc fonction propre est à chaque fois disponible pour l'élaboration des différents types de données.

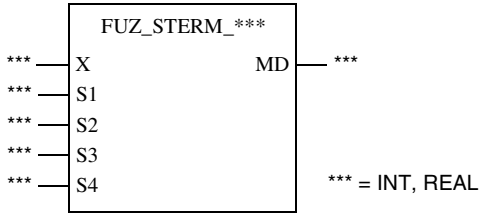
Le nombre des entrées peut être augmenté jusqu'à 4 au maximum en modifiant verticalement la taille du cadre du bloc fonction.

Comme paramètres supplémentaires, on peut configurer EN et ENO.

Représentation

Symbole

Représentation du bloc :



Description des paramètres

Description des paramètres du bloc :

Paramètre	Types de données	Signification
X	INT, REAL	Variable linguistique
S1	INT, REAL	Point d'appui S1
S2	INT, REAL	Point d'appui S2
:	:	:
S4	INT, REAL	Point d'appui S4
MD	INT, REAL	sortie degré d'appartenance (Membership Degree)

Description détaillée

Description des paramètres

Avec le bloc fonction FUZ_STERM, un terme d'une variable linguistique est fuzzifié. La fonction d'appartenance peut être définie par jusqu'à 4 points d'appui (S1 ... S4). Les points d'appui doivent être indiquées dans l'ordre croissant. Vous trouverez les fonctions standard dans le tableau suivant. Pour les cas spéciaux voir les diagrammes temporels. D'autres formes sont possibles en inversant l'ordre des points d'appui. Le tableau suivant présente les courbes de fonction possibles.

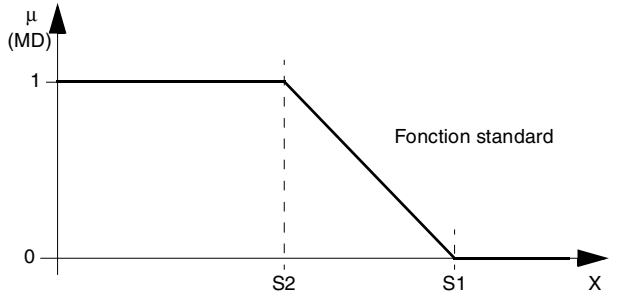
Fonctions standard

Fonction d'appartenance	Nombre d'entrées	Condition
Rampe descendante	2	S2<S1
Rampe ascendante	2	S1<S2
Triangle	3	S1<S2<S3
Trapèze	4	S1<S2<S3<S4

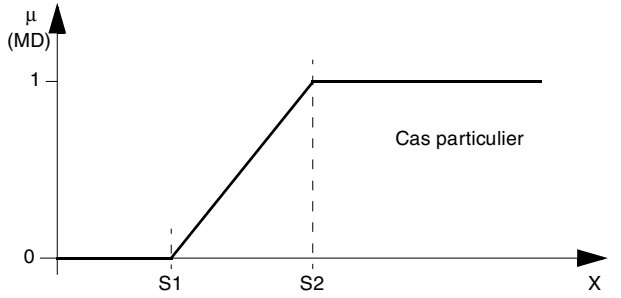
Diagramme de cycles

Bloc fonction avec 2 entrées (S1 .... S2)

rampe descendante : S2 < S1

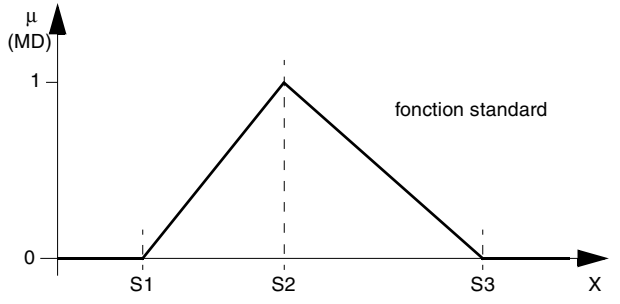


rampe ascendante : S1 < S2

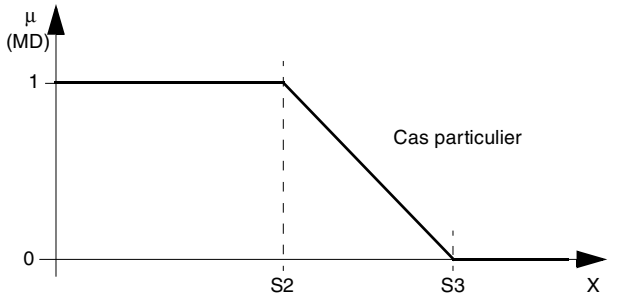


Bloc fonction avec 3 entrées (S1 .... S3)

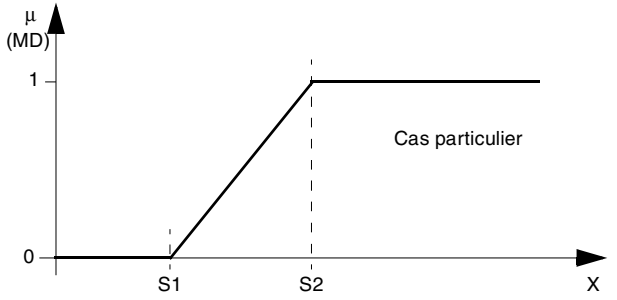
triangle : S1 < S2 < S3



rampe descendante : S2 < S3 und S1 > S2

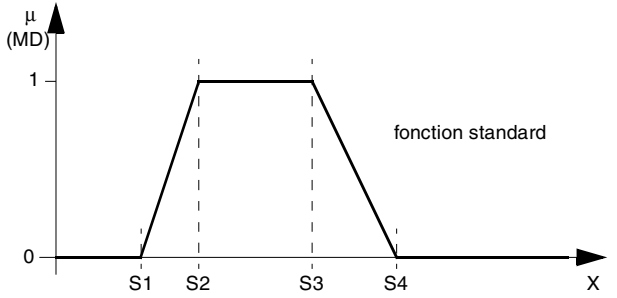


rampe ascendante : S1 < S2 und S3 < S2

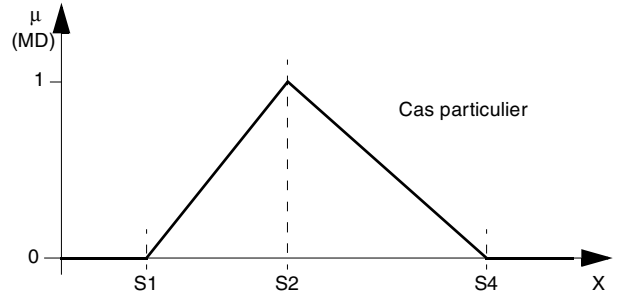


Bloc fonction avec 4 entrées(S1 .... S4)

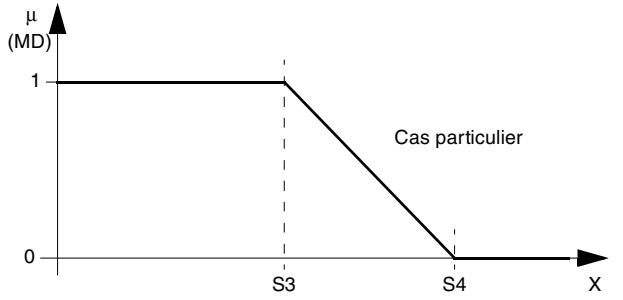
Trapèze : S1 < S2 < S3 < S4



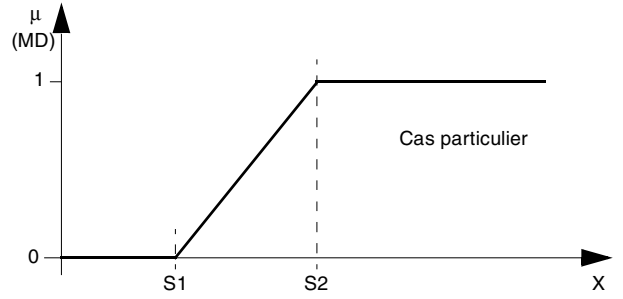
triangle : S1 < S2 < S4 et S3 ≤ S2



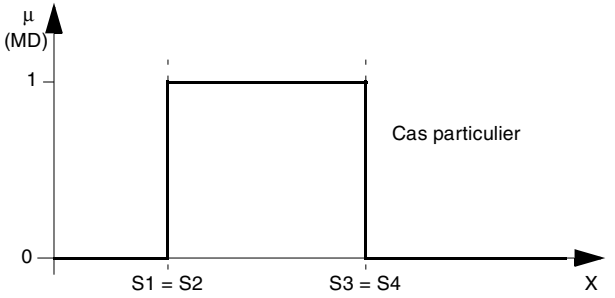
rampe descendante : S2 ≤ S3 < S4 und S1 > S2



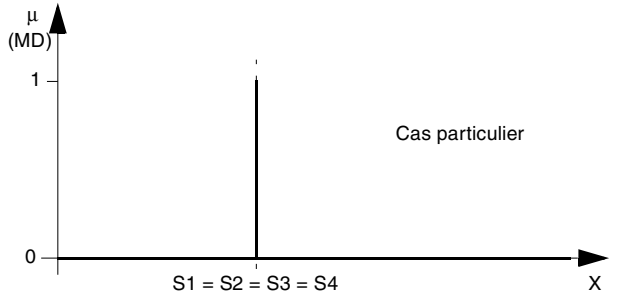
rampe ascendante : S1 < S2 ≤ S3 et S4 < S3



rectangle : S1 = S2 < S3 = S4



Singleton : S1 = S2 = S3 = S4



Erreur due au temps de transit

Erreur due au temps de transit

Un message d'erreur survient , lorsque

- le nombre des points d'appui > 4 est (E_EFB_TO MANY_INPUTS).

FUZ_SUM_*** : somme Fuzzy

11

Résumé

Introduction

Ce chapitre décrit les blocs FUZ_SUM_***.

Contenu de ce chapitre

Ce chapitre contient les sujets suivants :

Présentation

Description de la fonction

La fonction représente la somme limitée (sortie MD) du degré d'appartenance (entrées MD1 ... MDx). Les gammes de valeur des entrées et de la sortie comprend pour le type de données INT 0 ... à 10 000 et pour le type de données REAL 0 ... 1.

Le nombre des entrées peut être augmenté.

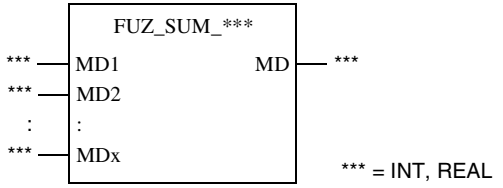
Les types de données de toutes les valeurs d'entrée et celles des valeurs de sortie doivent être identiques. Un bloc fonction propre est à chaque fois disponible pour l'élaboration des différents types de données.

Comme paramètres supplémentaires, on peut configurer EN et ENO.

Représentation

Symbole

Représentation du bloc :



Formule

Formule du bloc :

$$\text{REAL: MD} = \text{Min}\left\{1, \sum_{i=1}^{n} \text{MDi}\right\}$$

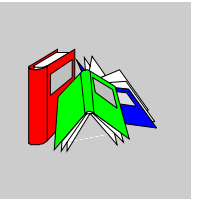
$$\text{INT: MD} = \text{Min}\left\{10000, \sum_{i=1}^{n} \text{MDi}\right\}$$

Description des paramètres

Description des paramètres du bloc :

Paramètre	Types de données	Signification
MD1	INT, REAL	1. degré d'appartenance (Membership Degree)
MD2	INT, REAL	2. degré d'appartenance (Membership Degree)
:	:	:
MDx	INT, REAL	x. degré d'appartenance (Membership Degree)
MD	INT, REAL	Sortie somme limitée

Glossaire



A

Abonné de réseau

Un abonné est un appareil avec une adresse (1 à 64) sur le réseau Modbus Plus.

Abonné local du réseau

L'abonné local est celui qui est projeté à l'instant.

Adresse abonné

L'adresse abonné sert à la désignation univoque d'un abonné du réseau dans l'itinéraire de routage. L'adresse est réglée directement sur l'abonné, p. ex. via le commutateur rotatif situé sur la face arrière du module.

Adresses

Les adresses (directes) sont des zones de mémoire dans l'API. Celles-ci se trouvent dans la mémoire d'état et peuvent être affectées à des modules d'entrée/sortie. L'affichage/la saisie d'adresses directes est possible dans les formats suivants :

- Format standard (400001)
- Format séparateur (4:00001)
- Format compact (4:1)
- Format CEI (QW1)

Affectation des E/S

L'affectation des E/S est une liste d'affectation générée à partir de la liste d'affectation de l'utilisateur. L'affectation des E/S est gérée dans l'API et contient p. ex. des informations sur l'état des stations et modules E/S, en supplément de la liste d'affectation de l'utilisateur.

ANL_IN ANL_IN est le type de données "entrée analogique" et est utilisé pour le traitement des valeurs analogiques. Les références 3x du module d'entrée analogique configuré déterminées dans la liste d'affectation des E/S sont affectées automatiquement au type de données et doivent de ce fait être occupées uniquement par des variables non localisées.

ANL_OUT ANL_OUT est le type de données "sortie analogique" et est utilisé pour le traitement des valeurs analogiques. Les références 4x du module de sortie analogique configuré déterminées dans la liste d'affectation des E/S sont affectées automatiquement au type de données et doivent de ce fait être occupées uniquement par des variables non localisées.

ANY Dans la présente version, "ANY" comprend les types de données élémentaires BOOL, BYTE, DINT, INT, REAL, UDINT, UINT, TIME et WORD ainsi que les types de données qui en sont dérivés.

ANY_BIT Dans la présente version, "ANY_BIT" comprend les types de données BOOL, BYTE et WORD.

ANY_ELEM Dans la présente version, "ANY_ELEM" comprend les types de données BOOL, BYTE, DINT, INT, REAL, UDINT, UINT, TIME et WORD.

ANY_INT Dans la présente version, "ANY_INT" comprend les types de données DINT, INT, UDINT et UINT.

ANY_NUM Dans la présente version, "ANY_NUM" comprend les types de données DINT, INT, REAL, UDINT et UINT.

ANY_REAL Dans la présente version, "ANY_REAL" correspond au type de données REAL.

API Automate programmable industriel

Appel La procédure par laquelle l'exécution d'une opération est lancée.

Argument Synonyme de paramètre réel.

Atrium L'automate basé sur PC est monté sur platine standard AT et s'utilise au sein d'un ordinateur hôte dans un emplacement de bus ISA. Ce module possède une carte mère (nécessite un pilote SA85) avec deux emplacements pour cartes filles PC104. L'une des cartes filles PC104 sert d'UC et l'autre à la commande INTERBUS.

Avertissement Si un état critique est identifié lors du traitement d'un FFB ou d'une étape (p. ex. des valeurs d'entrée critiques ou des limites temporelles dépassées), un avertissement est généré. Celui-ci peut être visualisé à l'aide de la commande **En ligne** → **Affichage événements...**. Sur les FFB, la sortie ENO reste sur "1".

B

Base de données de projet La base de données du PC, contenant les informations de configuration d'un projet.

Bibliothèque Ensemble d'objets logiciels prévus pour la réutilisation lors de la programmation de nouveaux projets, ou bien même pour l'élaboration de nouvelles bibliothèques. Les exemples sont les bibliothèques des types de blocs fonction élémentaires.
Les bibliothèques EFBpeuvent être subdivisées en groupes.

Bits d'entrée (Références 1x) L'état 1/0 des bits d'entrée est commandé par les données du procédé arrivant depuis un périphérique d'entrée dans l'UC.

Note : Le x suivant le premier chiffre du type de référence représente un emplacement à cinq chiffres dans la mémoire de données utilisateur, p. ex. la référence 100201 signifie un bit d'entrée à l'adresse 201 de la mémoire d'état.

Bits d'état Il existe un bit d'état pour chaque abonné à entrée globale, entrée ou sortie spécifique de données de diffusion. Si un groupe de données défini a pu être transmis avec succès avant écoulement du timeout réglé, le bit d'état correspondant est mis à 1. Dans le cas contraire, ce bit est mis à 0 et toutes les données appartenant à ce groupe (à 0) sont effacées.

Bits de sortie/ bits internes (Références 0x) Un bit de sortie/bit interne peut être utilisé pour commander des données de sortie réelles via une unité de sortie du système de contrôle, ou pour définir une ou plusieurs sorties TOR dans la mémoire d'état. Remarque : le x suivant immédiatement le premier chiffre du type de référence, représente un emplacement mémoire sur 5 chiffres dans la mémoire de données utilisateur, p. ex. la référence 000201 signifie un bit interne ou de sortie à l'adresse 201 de la mémoire d'état.

Bloc fonction (instance) (BF)

Un bloc fonction est une unité d'organisation de programme, qui, en fonction de sa fonctionnalité définie dans la description de type de bloc fonction, calcule des valeurs pour ses sorties et variable(s) interne(s), lorsqu'elle est appelée comme instance particulière. Toutes les valeurs des sorties et variables internes d'une instance particulière de bloc fonction sont conservées d'un appel du bloc fonction au suivant. Des appels répétés de la même instance de bloc fonction avec les mêmes arguments (valeurs des paramètres d'entrée) ne délivrent de ce fait pas forcément la (les) même(s) valeur(s) de sortie.
Chaque instance de bloc fonction est représentée graphiquement par un symbole rectangulaire. Le nom du type de bloc fonction est situé en haut au milieu, à l'intérieur du rectangle. Le nom de l'instance de bloc fonction est également en haut, bien qu'à l'extérieur du rectangle. Il est généré automatiquement à la création d'une instance mais peut, le cas échéant, être modifié par l'utilisateur. Les entrées sont représentées à gauche, les sorties à droite du bloc. Les noms des paramètres formels d'entrée/sortie sont indiqués à l'intérieur du rectangle aux places correspondantes.
La description ci-dessus de la représentation graphique est valable de principe également pour lesappels de fonction et pour les appels DFB. Les différences sont décrites dans les définitions correspondantes.

Bobine

Une bobine est un élément LD transmettant sans le modifier l'état de la liaison horizontale sur sa gauche à la liaison horizontale sur sa droite. L'état est alors mémorisé dans la variable/adresse directe associée.

BOOL

BOOL signifie type de données "booléen". La longueur des éléments de données est 1 bit (stocké en mémoire sur 1 octet). La plage de valeurs des variables de ce type de données est 0 (FALSE) et 1 (TRUE).

Bridge

Un bridge est un dispositif permettant de relier des réseaux. Il permet la communication entre abonnés de deux réseaux. Chaque réseau possède sa propre séquence de rotation de jeton - le jeton n'est pas transmis par les bridges.

BYTE

BYTE est le type de données "cordon de bits 8". L'entrée peut se faire en libellé en base 2, libellé en base 8 ou libellé en base 16. La longueur des éléments de données est de 8 bits. Il n'est pas possible d'affecter une plage de valeurs numériques à ce type de données.

C

CEI 611313

Norme internationale : Automates programmables Partie 3 : Langages de programmation.

Code de section	Le code de section est le code exécutable d'une section. La taille du code de section dépend principalement du nombre de blocs dans la section.
Code DFB	Le code DFB est le code DFB exécutable d'une section. La taille du code DFB dépend principalement du nombre de modules dans la section.
Code EFB	Le code EFB est le code exécutable de tous les EFB utilisés. Les EFB utilisés dans les DFB sont également pris en compte.
Configuration de transmission de données	Paramètres déterminant comment les informations sont transmises depuis votre PC vers l'API.
Connexion série	En connexion série (COM), les informations sont transmises bit par bit.
Constantes	Les constantes sont des variables non localisées, auxquelles est affectée une valeur qui ne peut être modifiée par la logique de programme (lecture seule).
Contact	Un contact est un élément LD transmettant un état sur la liaison horizontale située à sa droite. Cet état est le résultat d'une liaison ET booléenne entre l'état de la liaison horizontale sur sa gauche et l'état de la variable/adresse directe qui lui est affectée. Un contact ne modifie pas la valeur de la variable/adresse directe associée.
Convention CEI sur les noms (Identificateur)	Un identificateur est une suite de lettres, chiffres et caractères de soulignement devant commencer par une lettre ou un caractère de soulignement (p. ex. nom d'un type de bloc fonction, d'une instance, d'une variable ou d'une section). Les lettres des polices de caractères nationales (p. ex. : ö, ü, é, õ) peuvent être utilisées sauf dans les noms de projets et de DFB. Les caractères de soulignement sont significatifs dans les identificateurs ; p. ex. "A_BCD" et "AB_CD" seront interprétés comme des identificateurs différents. Plusieurs caractères de soulignement de tête ou de suite ne sont pas autorisés. Les identificateurs ne doivent pas comporter d'espaces. Les majuscules/minuscules ne sont pas significatives ; p. ex. "ABCD" et "abcd" seront interprétés comme le même identificateur. Les identificateurs ne doivent pas être des mots-clés.
Cordon de bits	C'est un élément de données constitué d'un ou de plusieurs bits.
Cycle programme	Un cycle programme consiste en la lecture des entrées, le traitement de la logique de programme et l'édition des sorties.

D

DDE (Echange dynamique de données)

L'interface DDE permet à deux programmes sous Windows d'échanger des données en dynamique. L'utilisateur peut se servir de l'interface DDE en moniteur étendu afin d'appeler ses propres applications d'affichage. Avec cette interface, l'utilisateur (c.-à-d. le client DDE) peut non seulement lire des données du moniteur étendu (le serveur DDE), mais peut également écrire des données sur l'API via le serveur. L'utilisateur peut ainsi modifier directement des données dans l'API tout en surveillant et en analysant les résultats. Lors de l'utilisation de cette interface, l'utilisateur peut créer son propre "Outil graphique", "Face Plate" ou "Outil de réglage", et intégrer celui-ci dans le système. Ces outils peuvent être écrits dans n'importe quel langage que le DDE prend en charge, p. ex. Visual Basic, VisualC++. Ils sont appelés lorsque l'utilisateur actionne l'un des boutons de commande de la boîte de dialogue Moniteur étendu. Outil graphique Concept : grâce au lien DDE entre Concept et l'outil Graphique Concept, il est possible de représenter les signaux d'une configuration sous forme de chronogramme.

Déclaration

Le mécanisme qui permet d'établir la définition d'un élément de langage. Normalement, une déclaration nécessite le rattachement d'un identificateur à l'élément de langage et l'affectation d'attributs, tels que lestypes de données et les algorithmes.

Défaut

Si, lors du traitement d'un FFB ou d'une étape, une erreur est détectée (p. ex. valeurs d'entrée non autorisées ou erreur de durée), un message d'erreur est généré, lequel peut être visualisé à l'aide de la commande **En ligne** → **Affichage événements...**. Sur les FFB la sortie ENOest mise à "0".

Défragmentation

La défragmentation permet de supprimer les trous indésirables dans la zone mémoire (générés, p. ex., en effaçant des variables inutilisées).

Derived Function Block (DFB) (Bloc fonction dérivé)

Un bloc fonction dérivé représente l'appel d'un type de bloc fonction dérivé. Vous trouverez des détails de la forme graphique de l'appel dans la définition "Bloc fonction (instance)". Contrairement aux appels de types d'EFB, les appels de types DFB sont caractérisés par des lignes verticales doubles sur les côtés gauche et droit du symbole rectangulaire du bloc.
Le corps d'un type de bloc fonction dérivé est projeté en langage FBD, langage LD, langage ST et langage IL quoique seulement dans la version actuelle du système de programmation. Les fonctions dérivées ne peuvent pas encore être définies dans la version actuelle.
On fait la distinction entre les DFB locaux et globaux.

DFB globaux Les DFB globaux sont disponibles dans tout projet Concept. Le stockage des DFB globaux dépend de la configuration dans le fichier CONCEPT.INI.

DFB locaux Les DFB locaux ne sont disponibles que dans un seul projet Concept et sont enregistrés dans le répertoire DFB sous le répertoire de projet.

Diagramme fonctionnel en séquence (SFC) Les éléments de langage SFC permettent de subdiviser une unité d'organisation de programme en un certain nombre d'étapes et de transitions, reliées entre elles par des liaisons dirigées. A chaque étape correspond un nombre d'actions et à chaque transition est associée une condition de transition.

DINT DINT signifie type de données "entier double (double integer)". L'entrée s'effectue en libellé entier, libellé en base 2, libellé en base 8 ou libellé en base 16. La longueur des éléments de données est de 32 bits. La plage de valeurs pour les variables de ce type de données va de -2 exp (31) à 2 exp (31) -1.

Données d'instance DFB Les données d'instance DFB sont des données internes des instructions chargeables dérivées utilisées dans le programme.

Données de section Les données de section sont les données locales d'une section, comme par ex. les libellés, les liaisons entre blocs, les entrées et sorties de bloc non liées, la mémoire d'état interne des EFB.

Note : Les données qui sont configurées dans les DFB de cette section ne sont pas des données de section.

Données globales Les données globales sont des variables non localisées.

DP (PROFIBUS) DP = Dezentrale Peripherie (périphérie décentralisée)

DX Zoom Cette caractéristique vous permet de vous raccorder sur un objet de programmation afin d'en surveiller des valeurs et de les modifier, si nécessaire.

E

Elément de langage Chaque élément de base dans l'un des langages de programmation CEI, p. ex. une étape en SFC, une instance de bloc fonction en FBD ou la valeur de départ d'une variable.

EN / ENO (autorisation / affichage d'erreur)

Si la valeur de EN vaut "0", lorsque le FFB est lancé, les algorithmes définis par le FFB ne sont pas exécutés et toutes les sorties conservent leur valeur précédente. La valeur de ENO est dans ce cas mise automatiquement à "0". Si la valeur de EN est "1" lors de l'appel du FFB, les algorithmes définis par le FFB seront exécutés. Après l'exécution sans erreur de ces algorithmes, la valeur de ENO est mise automatiquement à "1". Si une erreur survient lors de l'exécution de ces algorithmes, ENO est mis automatiquement à "0". Le comportement de sortie des FFB est indépendant du fait que ceux-ci sont appelés sans EN/ENO ou avec EN=1. Si l'affichage de EN/ENO est activé, l'entrée EN doit absolument être câblée. Le FFB n'est sinon jamais exécuté. L'activation/la désactivation de EN et ENO se fait dans la boîte de dialogue des caractéristiques du bloc fonction. Cette boîte de dialogue est appelée via **Objets** → **Propriétés...** ou en double-cliquant sur le FFB.

Erreur d'exécution

Erreur survenant lors du traitement du programme sur l'API sur des objets SFC (p. ex. des étapes) ou des FFB. Il s'agit p. ex. de dépassement de plage de valeurs sur les compteurs ou bien d'erreurs temporelles sur les étapes.

Etape

Elément de langage SFC : situation dans laquelle le comportement d'un programme suit, en fonction de ses entrées et sorties, les opérations définies par les actions correspondantes de l'étape.

Etape initiale (Etape de départ)

L'étape de démarrage d'une séquence. Une étape initiale doit être définie dans chaque séquence. La séquence est démarrée à son premier appel par l'étape initiale.

Evaluation

C'est le processus par lequel est déterminé une valeur d'une fonction ou des sorties d'un bloc fonction lors de l'exécution du programme.

Expression

Les expressions sont constituées d'opérateurs et d'opérandes.

F

Fenêtre active

Il s'agit de la fenêtre momentanément sélectionnée. Pour un instant donné, seule une fenêtre peut être active. Lorsqu'une fenêtre devient active, la couleur de sa barre de titre change afin de la distinguer des autres fenêtres. Les fenêtres non sélectionnées ne sont pas actives.

Fenêtre d'application

Il s'agit de la fenêtre contenant l'espace de travail, la barre de menus et la barre d'outils du programme applicatif. Le nom du programme applicatif apparaît dans la barre de titre. Une fenêtre d'application peut contenir plusieurs fenêtres de document. Dans Concept, la fenêtre d'application correspond à un projet.

Fenêtre de document — Une fenêtre contenue dans une fenêtre d'application. Plusieurs fenêtres de document peuvent être ouvertes simultanément dans une fenêtre d'application. Mais seule une fenêtre de document peut être active. Les fenêtres de document dans Concept sont p. ex. les sections, la fenêtre des messages, l'éditeur de données de référence et la configuration de l'automate.

FFB (fonctions/ blocs fonction) — Terme générique désignant les EFB (fonctions/blocs fonction élémentaires) et les DFB (blocs fonction dérivés)

Fichier de code source (Concept-EFB) — Le fichier de code source est un fichier source ordinaire en C++. Après exécution de la commande **Bibliothèque** → **Créer des fichiers**, ce fichier contient un cadre de code EFB dans lequel vous devez porter un code spécifique de l'EFB sélectionné. Pour ce faire, lancez la commande **Objets** → **Source**.

Fichier de définition (Concept-EFB) — Le fichier de définition contient des informations générales de description de l'EFB sélectionné et ses paramètres formels.

Fichier de sauvegarde (Concept-EFB) — Le fichier de sauvegarde est une copie du dernier fichier de code source. Le nom de ce fichier de sauvegarde est "backup??.c" (on suppose ce faisant que vous n'avez jamais plus de 100 copies de votre fichier de sauvegarde). Le premier fichier de sauvegarde porte le nom "backup00.c". Si vous avez procédé à des modifications dans le fichier de définition n'entraînant pas de modification d'interface pour l'EFB, vous pouvez vous dispenser de créer un fichier de sauvegarde en éditant son fichier de code source (**Objets** → **Source**). Si un fichier de sauvegarde est créé, vous pouvez lui donner le nom Fichiersource.

Fichier factice — Il s'agit d'un fichier vide constitué d'un en-tête contenant diverses informations générales sur le fichier, comme l'auteur, la date de création, la désignation de l'EFB, etc. L'utilisateur doit procéder à la préparation de ce fichier factice à l'aide d'entrées supplémentaires.

Fichier prototype (Concept-EFB) — Le fichier prototype contient tous les prototypes des fonctions affectées. On indique en outre, si elle existe, une définition type de la structure de la situation interne.

Fichier Template (Concept-EFB) — Le fichier Template est un fichier ASCII contenant des informations de mise en page pour l'éditeur FBD de Concept, ainsi que des paramètres pour la génération de code.

Filtre RIF — (Filtre Finite Impulse Response) Filtre à réponse impulsionnelle finie

Filtre RII — (Filtre Infinite Impulse Response) Filtre à réponse impulsionnelle infinie

Fonction (FUNK)

Une unité d'organisation de programme délivrant à l'exécution exactement un élément de donnée. Une fonction ne dispose pas d'information de situation interne. Les appels répétés de la même fonction avec les mêmes paramètres d'entrée délivrent toujours les mêmes valeurs de sortie.
Vous trouverez des détails de la forme graphique des appels de fonction dans la définition "Bloc fonction (instance)". Contrairement aux appels de blocs fonction, les appels de fonction ne disposent que d'une unique sortie sans nom, son nom étant le nom de la fonction elle-même. En FBD, chaque appel est caractérisé par un numéro unique par le bloc graphique ; ce numéro est créé automatiquement et ne peut pas être modifié.

Fonctions/blocs fonction élémentaires (EFB)

Caractérisation des fonctions ou des blocs fonction, dont les définitions de type n'ont pas été formulées dans l'un des langages CEI, c.-à-d. dont les corps p. ex. ne peuvent être modifiés à l'aide de l'éditeur DFB (Concept-DFB). Les types EFB sont programmés en "C" et sont mis à disposition en forme précompilée par les bibliothèques.

Format CEI (QW1)

Au début de l'adresse se trouve un identificateur conforme à CEI, suivi de l'adresse à cinq chiffres :

- %0x12345 = %Q12345
- %1x12345 = %I12345
- %3x12345 = %IW12345
- %4x12345 = %QW12345

Format compact (4:1)

Le premier chiffre (la référence) est séparé par deux points (:) de l'adresse suivante, les zéros de tête n'étant pas indiqués dans l'adresse.

Format séparateur (délimiteur) (4:00001)

Le premier chiffre (la référence) est séparé par deux-points (:) de l'adresse à cinq caractères.

Format standard (400001)

L'adresse à cinq positions se situe juste après le premier chiffre (la référence).

G

Groupes (EFB)

Quelques bibliothèques EFB (p. ex. la bibliothèque CEI) sont subdivisées en groupes. Cela simplifie, particulièrement dans les importantes bibliothèques, la recherche des EFB.

I

Instanciation La création d'une instance.

Instruction (IL) Les instructions sont des "commandes" du langage de programmation IL. Chaque instruction commence à une nouvelle ligne et est suivie d'un opérateur, le cas échéant avec modificateur, et, si nécessaire pour l'opération concernée, d'un ou de plusieurs opérandes. Si l'instruction utilise plusieurs opérandes, ceux-ci sont séparés par des virgules. Devant l'instruction peut se trouver une étiquette suivie de deux points. Le commentaire doit, s'il existe, être le dernier élément de la ligne.

Instruction (LL984) La mission d'un utilisateur lors de la programmation d'automatismes électriques est de mettre en oeuvre des instructions codées de façon opérationnelle sous forme d'objets imagés classés selon les formes identifiables de contact. Les objets du programme ainsi conçus sont convertis au niveau utilisateur en codes opérandes utilisables par l'ordinateur, et ce lors de la procédure de chargement. Les codes opérandes sont décodés dans l'UC et traités par les fonctions micrologicielles du contrôleur, de sorte que la commande désirée soit ainsi mise en oeuvre.

Instruction (ST) Les instructions sont des "commandes" du langage de programmation ST. Les instructions doivent se terminer par des points-virgules. Plusieurs instructions (séparées par des points-virgules) peuvent se trouver sur une même ligne.

INT INT correspond au type de données "nombre entier (integer)". L'entrée s'effectue en libellé entier, libellé en base 2, libellé en base 8 ou libellé en base 16. La longueur des éléments de données est de 16 bits. La plage de valeurs pour les variables de ce type de données va de -2 exp (15) à 2 exp (15) -1.

Interbus S (PCP) Afin d'utiliser le canal PCP de l'Interbus S et le prétraitement de données de procédé Interbus S (PDV), le configurateur Concept propose maintenant le nouveau type de station d'E/S Interbus S (PCP). A ce type de station d'E/S est affecté de manière fixe le module de connexion Interbus 180-CRP-660-01.
Le module 180-CRP-660-01 se distingue du 180-CRP-660-00 seulement par une plage d'E/S sensiblement plus importante dans la mémoire d'état de l'automate.

J

Jeton

Le jeton du réseau régit la possession momentanée du droit de transmission d'un abonné individuel. Le jeton circule entre les abonnés dans un sens circulaire (croissant) des adresses. Tous les abonnés suivent la rotation du jeton et peuvent obtenir toute sorte de données qui y sont véhiculées.

L

Langage en blocs fonctionnels (FBD)

Une ou plusieurs sections contenant des réseaux représentés graphiquement composés de fonctions, blocs fonction et liaisons.

Liaison

Une liaison de contrôle ou de données entre objets graphiques (p. ex. étapes dans l'éditeur SFC, blocs fonction dans l'éditeur FBD) au sein d'une section, graphiquement représenté par une ligne.

Liaison locale (Local Link)

La liaison locale de réseau est le réseau reliant l'abonné local à d'autres abonnés, soit directement soit par l'amplificateur de bus.

Liaisons binaires

Il s'agit de liaisons entre des sorties et des entrées de FFB de type de données BOOL.

Libellé

Les libellés servent à fournir des valeurs directement aux entrées des FFB, conditions de transition etc... Ces valeurs ne peuvent pas être écrasées par la logique du programme (lecture seule). Le système distingue les libellés génériques des libellés classés par type.
De plus, les libellés servent à affecter une valeur à une constante ou une valeur initiale à une variable.
L'entrée se fait en libellé en base 2, libellé en base 8, libellé en base 16, libellé entier, libellé réel ou libellé réel avec exposant.

Libellé de durée

Les unités permises pour les durées (TIME) sont les jours (J), les heures (H), les minutes (M), les secondes (S) et les millisecondes (MS) ou une combinaison de ceux-ci. La durée doit être caractérisée par le préfixe t#, T#, time# ou TIME#. Le "dépassement" de l'unité de plus grande valeur est admise; p. ex. l'entrée T#25H15M est permise.

Exemple
t#14MS, T#14.7S, time#18M, TIME#19.9H, t#20.4D, T#25H15M, time#5D14H12M18S3.5MS

Libellé en base 16

Les libellés en base 16 servent à codifier les entiers dans le système hexadécimal. La base doit être repérée par le préfixe 16#. Les valeurs doivent être non signées (+/). Les caractères de soulignement individuels (_) entre les chiffres ne sont pas significatifs.

Exemple
16#F_F ou 16#FF (décimal 255)
16#E_0 ou 16#E0 (décimal 224)

Libellé en base 2

Les libellés en base 2 servent à la codification de valeurs entières dans le système de base 2. La base doit être repérée par le préfixe 2#. Les valeurs doivent être non signées (+/). Les caractères de soulignement individuels (_) entre les chiffres ne sont pas significatifs.

Exemple
2#1111_1111 ou 2#11111111 (255 décimal)
2#1110_0000 ou 2#11100000 (224 décimal)

Libellé en base 8

Les libellés en base 8 servent à codifier les entiers dans le système de base 8. La base doit être repérée par le préfixe 8#. Les valeurs doivent être non signées (+/). Les caractères de soulignement individuels (_) entre les chiffres ne sont pas significatifs.

Exemple
8#3_77 ou 8#377 (255 décimal)
8#34_0 ou 8#340 (décimal 224)

Libellé entier

Les libellés entiers servent à indiquer des valeurs entières dans le système décimal. Les valeurs peuvent être signées (+/). Les caractères de soulignement individuels (_) entre les chiffres ne sont pas significatifs.

Exemple
-12, 0, 123_456, +986

Libellés classés par type

Si vous voulez déterminer le type de données d'un libellé, vous pouvez le faire avec la construction suivante : 'nomtypedonnée'#'Valeur du libellé'

Exemple
INT#15 (type de données : entier, valeur : 15),
BYTE#00001111 (type de données : octet, valeur : 00001111)
REAL#23.0 (type de données : réel, valeur : 23,0)

Pour l'affectation du type de données REAL, vous pouvez indiquer la valeur de la manière suivante : 23.0.
En indiquant ce point décimal, le type de données REAL est affecté automatiquement.

Libellés génériques

Si le type de données d'un libellé n'a pas d'importance pour vous, indiquez la valeur du libellé. Dans ce cas, Concept affecte automatiquement un type de données adéquat au libellé.

Libellés réels

Les libellés réels servent à indiquer les valeurs à virgule flottante dans le système décimal. Les libellés réels s'identifient au point décimal. Les valeurs peuvent être signées (+/). Les caractères de soulignement individuels (_) entre les chiffres ne sont pas significatifs.

Exemple
-12.0, 0.0, +0.456, 3.14159_26

Libellés réels avec exposant

Les libellés réels avec exposant servent à indiquer les valeurs à virgule flottante dans le système décimal. Les libellés réels avec exposant se caractérisent par le point décimal. L'exposant donne la puissance de dix avec lequel le chiffre de devant doit être multiplié pour obtenir la valeur à représenter. La base peut être précédée d'un signe moins (). L'exposant peut être signé (+/-). Les caractères de soulignement individuels (_) entre les chiffres ne sont pas significatifs. (Uniquement entre les chiffres, et non avant ou après la virgule ou avant ou après "E", "E+" ou "E-")

Exemple
-1.34E-12 ou -1.34e-12
1.0E+6 ou 1.0e+6
1.234E6 ou 1.234e6

Liste d'affectation des E/S

Dans la liste d'affectation des E/S, on configure les modules d'E/S et modules experts des différentes unités centrales.

Liste d'instructions (IL)

IL est un langage littéral conforme à la norme CEI 1131, dans lequel les opérations, telles que les appels sur ou sans condition de blocs fonction et de fonctions, les sauts conditionnels ou sans condition, etc., sont représentées par des instructions.

Littéral structuré (ST)

ST est un langage littéral conforme à la CEI 1131, dans lequel les opérations, comme le lancement de blocs fonction et de fonctions, les exécutions conditionnelles d'instructions, la réitération d'instructions, etc. sont représentés par des instructions.

M

Macro

Les macros sont créées à l'aide du logiciel Concept-DFB.
Les macros servent à dupliquer des sections et des réseaux fréquemment utilisés (y compris leur logique, leurs variables et leur déclaration de variable).
On fait la distinction entre les macros locales et globales.

Les macros possèdent les caractéristiques suivantes :

- Les macros ne peuvent être créées qu'avec les langages FBD et LD
- Les macros ne contiennent qu'une seule section
- Elles peuvent contenir une section d'une complexité quelconque
- D'un point de vue programme, une macro instanciée, c.-à-d. une macro insérée dans une section, ne se distingue pas d'une section créée de manière conventionnelle.
- Appel de DFB dans une macro
- Déclaration de variables
- Utilisation de structures de données propres aux macros
- Validation automatique des variables déclarées dans la macro
- Valeurs initiales des variables
- Instanciation multiple d'une macro dans tout le programme avec différentes variables
- Le nom de la section, les noms des variables et le nom de la structure de données peuvent comporter jusqu'à 10 marques d'échange (@0 à @9) différentes.

Macros globales

Les macros globales sont disponibles dans tout projet Concept et sont enregistrées dans le répertoire DFB directement situé sous le répertoire Concept.

Macros locales

Les macros locales ne sont disponibles que dans un seul projet Concept et sont enregistrées dans le répertoire DFB sous le répertoire de projet.

Mémoire d'état La mémoire d'état est l'emplacement mémoire pour toutes les grandeurs sollicitées dans le programme utilisateur par des références (représentation directe). Par exemple les bits d'entrée, les bits de sortie/bits internes, les mots d'entrée et mots de sortie/mots internes se trouvent en mémoire d'état.

Mémoire du programme CEI La mémoire du programme CEI comprend le code programme, le code EFB, les données de section et les données d'instance DFB.

MMI Interface Homme-Machine

Mode ASCII American Standard Code for Information Interchange. Le mode ASCII est utilisé pour la communication avec différents équipements hôte. ASCII fonctionne sur 7 bits de données.

Mode RTU Remote Terminal Unit
Le mode RTU est utilisé pour la communication entre l'API et un ordinateur personnel compatible IBM. RTU fonctionne sur 8 bits de données.

Module SA85 Le module SA85 est une carte Modbus Plus pour ordinateur IBM-AT ou compatible.

Mots d'entrée (Références 3x) Un mot d'entrée contient des informations émanant d'une source externe et par lesquelles un nombre sur 16 bits est représenté. Un registre 3x peut également contenir 16 bits successifs lus dans le registre au format binaire ou BCD (binaire codé décimal). Remarque : le x suivant immédiatement le premier chiffre du type de référence, représente un emplacement mémoire à cinq chiffres dans la mémoire de données utilisateur, p.ex. la référence 300201 signifie un mot d'entrée de 16 bits à l'adresse 201 de la mémoire d'état.

Mots de sortie/ mots internes (Références 4x) Un mot de sortie/mot interne peut être utilisé pour la mémorisation de données numériques (binaires ou décimales) en mémoire d'état, ou bien pour envoyer des données depuis l'UC vers une unité de sortie du système de contrôle. Remarque : le x suivant immédiatement le premier chiffre du type de référence, représente un emplacement mémoire à cinq chiffres dans la mémoire de données utilisateur, p.ex. la référence 400201 signifie un mot de sortie/mot interne de 16 bits à l'adresse 201 de la mémoire d'état.

Mots-clés Les mots-clés sont des combinaisons uniques de caractères utilisés comme éléments spéciaux de syntaxe comme il est défini à l'annexe B de la CEI 1131-3. Tous les mots-clés utilisés dans la CEI 1131-3 et donc dans Concept, sont listés en annexe C de la CEI 1131-3. Ces mots-clés répertoriés ne doivent être utilisés à aucune autre fin, p. ex. pas comme nom de variable, nom de section, nom d'instance, etc.

N

Node

Un node est une cellule de programmation dans un réseau LL984. Une cellule/un node comprend une matrice 7x11, c.-à-d. 7 lignes de 11 éléments.

Nom d'étape

Le nom d'étape sert à la désignation unique d'une étape dans une unité d'organisation de programme. Le nom d'étape est créé automatiquement, mais peut être édité. Il doit être unique dans toute l'unité d'organisation de programme, sinon un message d'erreur apparaît.
Le nom d'étape créé automatiquement a toujours la structure suivante : S_n_m

S = Etape
n = Numéro de la section (numéro courant)
m = Numéro de l'étape dans la section (numéro courant)

Nom d'instance

Un identificateur, associé à une instance spécifique de bloc fonction.. Le nom d'instance sert au repérage sans univoque d'un bloc fonction au sein d'une unité d'organisation de programme. Le nom d'instance est créé automatiquement, mais peut être édité. Le nom d'instance doit être unique dans toute l'unité d'organisation de programme, la distinction Majuscule/Minuscule n'est pas faite. Si le nom saisi existe déjà, vous en êtes averti et vous devez choisir un autre nom. Le nom d'instance doit satisfaire aux conventions de noms CEI, sinon un message d'erreur apparaît. Le nom d'instance créé automatiquement a toujours la structure suivante : FBI_n_m

FBI = Instance de bloc fonction
n = Numéro de la section (numéro courant)
m = Numéro de l'objet FFB dans la section (numéro courant)

Numéro d'identification

Le numéro d'identification sert à caractériser de manière unique une fonction dans un programme ou DFB. Le numéro d'identification ne peut être édité et est attribué automatiquement. Il a toujours la structure : .n.m

n = Numéro de la section (numéro courant)
m = Numéro de l'objet FFB dans la section (numéro courant)

O

Opérande

Un opérande est un libellé, une variable, un appel de fonction ou une expression.

Opérateur Un opérateur est un symbole d'une opération arithmétique ou booléenne à exécuter.

P

Paramètre d'entrée (Entrée) Transmet lors de l'appel d'un FFB l'argument s'y rapportant.

Paramètre de sortie (Sortie) Un paramètre avec lequel est (sont) retourné(s) le(s) résultat(s) de l'évaluation d'un FFB.

Paramètre réel Paramètre d'entrée/sortie actuellement attribué.

Paramètres formels Paramètres d'entrée/sortie, utilisés au sein de la logique d'un FFB et sortant du FFB en entrées ou en sorties.

Paysage Le format paysage signifie que la page, au regard du texte imprimé, est plus large que haute.

PC Le matériel et le logiciel gérant (supportant) la programmation, l'élaboration, le test, la mise en service et la recherche de défauts dans les applications API ainsi que dans les applications système décentralisées, afin de rendre possible la documentation et l'archivage des sources. Le cas échéant, le PC peut également être utilisé pour la visualisation du procédé.

Portrait Portrait signifie que la page, au regard du texte imprimé, est plus haute que large.

Presse-papiers Le presse-papiers est une mémoire temporaire pour les objets coupés ou copiés. Ces objets peuvent être collés dans des sections. A chaque nouveau "couper" ou "copier", l'ancien contenu du presse-papiers est écrasé.

Processeur de communication Le processeur de communication traite les passages de jeton et le flux de données entre le réseau Modbus Plus et la logique utilisateur de l'API.

Programmation de la redondance d'UC (Hot Standby) Un système redondant est constitué de deux API configurés de manière identique qui communiquent entre eux à l'aide de processeurs redondants. En cas de panne de l'API primaire, l'API secondaire prend le contrôle de l'automatisme. Dans les conditions normales, l'API secondaire n'effectue aucune fonction de commande mais il vérifie les informations d'état afin de déceler les erreurs.

Programme La plus haute unité d'organisation de programme. Un programme est chargé en entier sur un seul API.

Projet

Appellation générale du niveau le plus élevé d'une arborescence logicielle, qui définit le nom de projet supérieur d'une application d'API. Après avoir défini le nom du projet, vous pouvez sauvegarder votre configuration système et votre programme de commande sous ce nom. Toutes les données apparaissant lors de la création de la configuration et du programme font partie de ce projet supérieur pour cette tâche spéciale d'automatisation.
Désignation générale du jeu complet d'informations de programmation et de configuration dans la base de données de projet, laquelle représente le code source décrivant l'automatisation d'une installation.

R

REAL

REAL correspond au type de données "nombre à virgule flottante". L'entrée se fait en libellé réel ou en libellé réel avec exposant. La longueur des éléments de données est de 32 bits. Plage des valeurs des variables de ce type de données : +/-3.402823E+38.

Note : En fonction du type de processeur mathématique de l'UC, différentes zones de cette plage de valeurs permise ne peuvent pas être affichées. Cela s'applique aux valeurs tendant vers ZERO et aux valeurs tendant vers l'INFINI. Dans ces cas, une valeur NAN (**N**ot **A N**umber) ou INF (**INF**inite (infini)) est affichée en mode Animation.

Référence

Toute adresse directe est une référence commençant par un code indiquant s'il s'agit d'une entrée ou d'une sortie et s'il s'agit d'un bit ou d'un mot. Les références commençant par le chiffre 6 représentent des registres de la mémoire étendue de la mémoire d'état.
Plage 0x = bits internes/de sortie
Plage 1x = bits d'entrée
Plage 3x = mots d'entrée
Plage 4x = mots internes/de sortie
Plage 6x = registres dans la mémoire étendue

Note : Le x suivant immédiatement le premier chiffre de chaque type de référence représente un emplacement mémoire à cinq chiffres dans la mémoire de données utilisateur, p.ex. la référence 400201 signifie un mot de sortie/mot interne de 16 bits à l'adresse 201 de la mémoire d'état.

Registres dans la mémoire étendue (référence 6x)

Les références 6x sont des mots indicateurs dans la mémoire étendue de l'API. Ils ne peuvent être utilisés que pour les programmes utilisateur LL984 et seulement sur les UC CPU 213 04 ou CPU 424 02.

Représentation directe

Une méthode pour représenter une variable dans un programme d'API, à partir de laquelle peut être déterminée directement une correspondance avec un emplacement logique, et indirectement avec l'emplacement physique.

Réseau

Un réseau est une connexion commune d'appareils sur une voie de données commune qui communiquent entre eux à l'aide d'un protocole commun.

Réseau décentralisé (DIO)

Une programmation décentralisée dans le réseau Modbus Plus permet une performance maximale de l'échange de données et n'a aucune exigence particulière sur les liaisons. La programmation d'un réseau décentralisé est simple. La configuration du réseau ne nécessite pas de logique de schéma à contacts supplémentaire. Toutes les conditions du transfert de données sont remplies en renseignant les paramètres correspondants du processeur de communication.

RIO (E/S décentralisée)

L'E/S décentralisée indique un emplacement physique des appareils E/S à commande par point par rapport au processeur qui les gère. Les entrées/sorties décentralisées sont reliées avec l'appareil de commande via un câble de communication.

S

Saut

Elément du langage SFC. Les sauts sont utilisés pour éviter des zones de la séquence.

Schéma à contacts (LD)

Le schéma à contacts est un langage de programmation graphique conforme à la CEI1131, dont l'aspect visuel suit les "échelons" d'un schéma à relayage.

Schéma à contacts 984 (LL)

Comme leur nom l'indique, les schémas à contacts comportent des contacts. Contrairement à un schéma électrique, les électrotechniciens se servent d'un schéma à contacts pour dessiner un circuit (à l'aide de symboles électriques). Celui-ci doit montrer l'évolution d'événements, et non les fils en présence qui relient les différentes parties entre elles. Une interface de schéma à contacts permet de réaliser une interface utilisateur traditionnelle pour commander les actions des constituants d'automatisme, afin que les électrotechniciens ne soient pas obligés d'apprendre un langage de programmation avec lequel ils ne seraient pas à l'aise. La construction d'un schéma à contacts effectif permet de relier des éléments électriques de manière à créer une sortie de commande. Celle-ci dépend d'un flux d'énergie logique passant par les objets électriques utilisés, lesquels représentent la condition préalable nécessaire d'un appareil électrique physique.
Sous une forme simple, l'interface utilisateur est un écran vidéo élaboré par l'application de programmation d'API, organisant un quadrillage vertical et horizontal dans lequel sont rangés des objets de programmation. Le schéma reçoit du courant par le côté gauche du quadrillage, et par connexion à des objets activés, le courant circule de gauche à droite.

Section

Une section peut par exemple être utilisée pour décrire le principe de fonctionnement d'une unité technologique telle qu'un moteur.
Un programme ou un DFB est constitué d'une ou de plusieurs sections. Les sections peuvent être programmées à l'aide des langages de programmation CEI FBD et SFC. Au sein d'une même section, seul un des langages de programmation mentionnés peut être utilisé.
Dans Concept, chaque section a sa propre fenêtre de document. Cependant, pour des raisons de clarté, il est conseillé de subdiviser une grande section en plusieurs petites. La barre de défilement sert à se déplacer au sein d'une section.

Station d'E/S DCP

A l'aide d'un processeur de contrôle distribué (D908), vous pouvez configurer un réseau décentralisé piloté par un API. Lorsque l'on utilise un D908 avec API décentralisé, l'API pilote considère l'API décentralisé comme une station d'E/S décentralisée. Le D908 et l'API décentralisé communiquent par le bus système, ce qui permet une grande performance pour un effet minimal sur le temps de cycle. L'échange de données entre le D908 et l'API pilote s'effectue par le bus d'E/S décentralisé à 1,5 Mégabit par seconde. Un API pilote peut gérer jusqu'à 31 processeurs D908 (adresse 2-32).

SY/MAX

Dans les automates Quantum, Concept gère la mise à disposition des modules d'E/S SY/MAX sur l'affectation des E/S pour la commande RIO par l'API Quantum. Le châssis distant SY/MAX dispose d'une carte d'E/S distante à l'emplacement 1, laquelle communique par un système d'E/S Modicon S908 R. Les modules d'E/S SY/MAX vous sont listés pour la sélection et la prise en compte dans l'affectation des E/S de la configuration Concept.

Symbole (icône) Représentation graphique de différents objets sous Windows, p. ex. lecteurs, programmes utilisateur et fenêtre de document.

T

Tas CEI Le tas CEI comprend la mémoire du programme CEI et les données globales.

TIME TIME est le type de données "durée". L'entrée se fait sous forme de libellé de durée. La longueur des éléments de données est de 32 bits. La plage de valeurs des variables de ce type de données va de 0 à 2exp(32)-1. L'unité du type de données TIME est 1 ms.

Transition La condition par laquelle la commande d'une ou de plusieurs étapes précédentes passe à une ou plusieurs étapes suivantes le long d'une liaison.

Type de bloc fonction Un élément de langage constitué de : 1. la définition d'une structure de données, subdivisée en variables d'entrée, de sortie et internes ; 2. un jeu d'opérations exécutées avec les éléments de la structure de données, lorsqu'une instance du type de bloc fonction est appelée. Ce jeu d'opérations peut être formulé soit dans l'un des langages CEI (type DFB) ou en "C" (type EFB). Un type de bloc fonction peut être instancié (appelé) plusieurs fois.

Type de données dérivé Les types de données dérivés sont des types de données qui ont été dérivés des types de données élémentaires et/ou d'autres types de données dérivés. La définition des types de données dérivés s'effectue dans l'éditeur de type de données de Concept.
On fait la distinction entre les types de données globaux et les types de données locaux.

Type de données générique Un type de données représentant plusieurs autres types de données.

Types de données

La vue d'ensemble montre la hiérarchie des types de données et comment ils sont utilisés aux entrées et sorties des fonctions et blocs fonction. Les types de données génériques sont caractérisés par le préfixe "ANY".

- ANY_ELEM
  - ANY_NUM
    ANY_REAL (REAL)
    ANY_INT (DINT, INT, UDINT, UINT)
  - ANY_BIT (BOOL, BYTE, WORD)
  - TIME
- Types de données système (Extension CEI)
- Dérivé (des types de données 'ANY')

Types de données dérivés globaux

Les types de données dérivés globaux sont disponibles dans tout projet Concept et sont enregistrés dans le répertoire DFB directement situé sous le répertoire Concept.

Types de données dérivés locaux

Les types de données dérivés locaux ne sont disponibles que dans un seul projet Concept et ses DFB locaux et sont enregistrés dans le répertoire DFB sous le répertoire de projet.

U

UDEFB

Fonctions/Blocs fonction élémentaires défini(e)s par l'utilisateur
Fonctions ou blocs fonction créés en langage de programmation C et que Concept met à votre disposition dans des bibliothèques.

UDINT

UDINT représente le type de données "entier double non signé (unsigned double integer)". L'entrée s'effectue en libellé entier, libellé en base 2, libellé en base 8 ou libellé en base 16. La longueur des éléments de données est de 32 bits. La plage de valeurs des variables de ce type de données va de 0 à 2exp(32)-1.

UINT

UINT représente le type de données "entier non signé (unsigned integer)". L'entrée s'effectue en libellé entier, libellé en base 2, libellé en base 8 ou libellé en base 16. La longueur des éléments de données est de 16 bits. La plage des valeurs des variables de ce type de données va de 0 à 2 exp(16) -1.

Unité d'organisation de programme

Une fonction, un bloc fonction ou un programme. Ce terme peut se rapporter à un type ou à une instance.

V

Valeur initiale

La valeur affectée à une variable lors du lancement du programme. L'affectation de la valeur s'effectue sous forme d'un libellé.

Variable localisée

Une adresse de mémoire d'état (adresses de références 0x, 1x, 3x, 4x) est affectée aux variables localisées. La valeur de ces variables est enregistrée dans la mémoire d'état et peut être modifiée en ligne au moyen de l'éditeur de données de référence. Ces variables peuvent être adressées avec leur nom symbolique ou avec leur adresse de référence.

Toutes les entrées et les sorties de l'API sont reliées à la mémoire d'état. L'accès du programme aux signaux des périphériques connectés à l'API ne se fait que via des variables localisées. Les accès de l'extérieur via les interfaces Modbus ou Modbus Plus de l'API, p. ex. des systèmes de visualisation, sont également possibles via des variables localisées.

Variable non localisée

Aucune adresse de mémoire d'état n'est affectée aux variables non localisées. Elles n'occupent donc pas non plus d'adresse de mémoire d'état. La valeur de ces variables est enregistrée dans le système et peut être modifiée en ligne au moyen de l'éditeur de données de référence. Ces variables ne sont adressées que par leur nom symbolique.

Les signaux ne disposant pas d'accès à la périphérie, p. ex, résultats intermédiaires, repères systèmes, etc., doivent être de préférence déclarés comme variable non localisée.

Variables

Les variables servent à l'échange de données au sein de sections, entre plusieurs sections et entre le programme et l'API.
Les variables consistent au moins en un nom de variable et un type de données. Si une adresse directe (référence) est affectée à une variable, on parle alors de variable localisée. Si aucune adresse directe n'est affectée à une variable, on parle alors de variable non localisée. Si un type de données dérivé est affecté à une variable, on parle alors d'une variable multi-éléments.
Il existe en outre des constantes et des libellés.

Variables de tableau

Variables auxquelles sont affectées untype de données dérivé défini à l'aide du mot clé ARRAY (tableau). Un tableau est un ensemble d'éléments de données appartenant au même type.

Variables multi-éléments

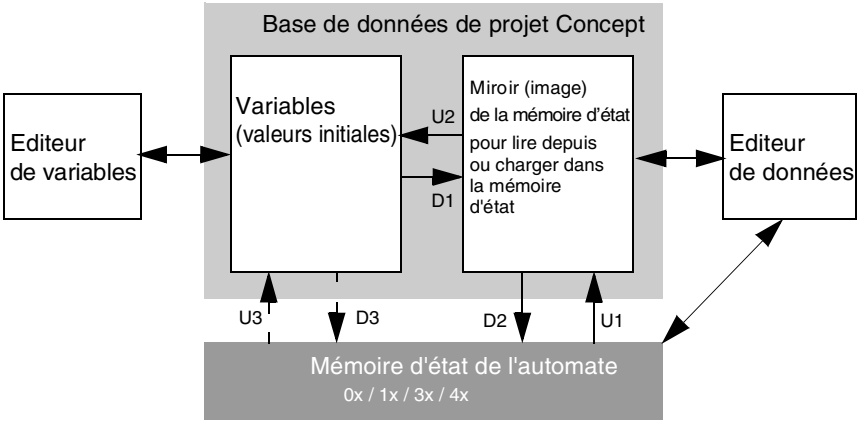
Variables, auxquelles est affecté un type de données dérivé défini avec STRUCT ou ARRAY.
On fait ici la distinction entre variables de tableau et variables structurées.

Variables structurées

Variables auxquelles est affecté un type de données dérivé défini avec STRUCT (structure).
Une structure est un ensemble d'éléments de données avec en général différents types de données (types de données élémentaires et/ou types de données dérivés).

Vue d'ensemble de la mémoire d'état lors de la lecture et du chargement

Vue d'ensemble :

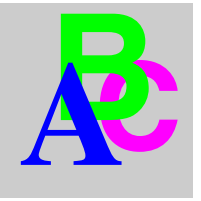


W

WORD

WORD correspond au type de données "Cordon de bits 16". L'entrée peut se faire en libellé en base 2, libellé en base 8 ou libellé en base 16. La longueur des éléments de données est de 16 bits. Il n'est pas possible d'affecter une plage de valeurs numériques à ce type de données.

Index



I

O

P

R

S

T

V